ulm university universität
uulm

Datenflussvarianten in Prozessmodellen: Szenarien, Herausforderungen, Ansätze

Stefanie Kaiser, Manfred Reichert

Ulmer Informatik-Berichte

Nr. 2011-03
Mai 2011

Datenflussvarianten in Prozessmodellen: Szenarien, Herausforderungen, Ansätze

Stefanie Kaiser, Manfred Reichert
Institut für Datenbanken und Informationssysteme, Universität Ulm

Zusammenfassung

Die geeignete Unterstützung von Geschäftsprozessen gewinnt durch die Abkehr vom funktionalen betrieblichen Aufbau hin zu prozessorientierten Organisationsstrukturen zunehmend an Bedeutung. Dabei wurde erkannt, dass ein bestimmter Prozesstyp oftmals nicht nur durch ein einziges Prozessmodell beschrieben werden kann, sondern dass abhängig vom Anwendungskontext unterschiedliche Prozessvarianten existieren. Die Gründe, mehrere Prozessvarianten gleichzeitig zu unterstützen und nicht auf einer einzigen Definition zu beharren, sind vielfältig. So müssen z.B. bei internationalen Unternehmen die Prozesse an die Rahmenbedingungen des jeweiligen Landes (z.B. gesetzliche Regularien) oder eine prozessorientierte Software für den jeweiligen Anwendungskontext angepasst werden. Der Umgang Prozessvarianten wurde in verschiedenen Forschungsarbeiten thematisiert, wobei diese mehrheitlich auf die *Kontrollflussperspektive* fokussieren. Die *Datenflussperspektive* ist im Zusammenhang mit Prozessvarianten noch wenig beachtet. Aus diesem Grund soll im Folgenden untersucht werden, wie das Management von Prozessvarianten unter dem Gesichtspunkt eines modifizierten Datenflusses gestaltet werden sollte. Es wird analysiert, unter welchen Bedingungen der Datenfluss eine Änderung erfährt und welchen Anforderungen ein Prozessvariantenmodellierungsansatz genügen muss.

Schlüsselworte: Prozessvarianten, Prozessmodellierung, Datenfluss

1 Einleitung

Geschäftsprozesse sind Abfolgen von Aktivitäten. Sie können auf verschiedenen Aggregationsebenen betrachtet werden und dienen dazu, eine Wertschöpfung zu erzielen [Gab10]. Geschäftsprozessen steht gemäß dieser Definition eine oder mehrere Prozesseingaben zur Verfügung, die durch sie zu „einem Kundennutzen stiftenden Output“ führen. An dieser Definition lassen sich drei integrale Bestandteile von Geschäftsprozessen identifizieren. An erster Stelle ist der Kontrollfluss zu nennen, anhand dessen die Abfolge und die Ausführungsbedingungen der Aktivitäten festgelegt werden. Der zweite Bestandteil fokussiert auf das Speisen und das Produkt eines Prozesses. Um eine Eingabe in ein Ergebnis zu transformieren, fließt diese entlang des Geschäftsprozesses und wird verändert. Beispielsweise werden Materialien durch einen Geschäftsprozess zu einem Produkt verarbeitet. Dies wird für den Geschäftsprozess mithilfe von Daten dokumentiert. Deshalb kann an zweiter Stelle der Datenfluss als integraler Bestandteil eines Geschäftsprozesses betrachtet werden. Schließlich sind die beteiligten Rollen zu nennen. Dabei ist zwischen Rollen innerhalb einer Firma und Rollen von externen Beteiligten wie Lieferanten oder Kunden zu unterscheiden.
In Unternehmen existiert eine Vielzahl von Geschäftsprozessen [MHHR06, LM07, MRB08]. Dabei ist festzustellen, dass auch Prozessvarianten auftreten können [SOS05, Hal09, WW10]. Varianten

zeichnen sich dadurch aus, dass sie ein und derselben Zielerreichung dienen, sie aber zumindest in Teilen unterschiedlich modelliert sind. Sie dienen dazu, den spezifischen Gegebenheiten wie rechtlichen Bestimmungen eines Landes oder Organisationsstrukturen von Abteilungen, gerecht zu werden. Verschiedene Forschungsansätze thematisieren das *Variantenmanagement*: Entweder wird davon ausgegangen, dass voneinander unabhängig modellierte Varianten existieren, die anhand ihrer ähnlichen Struktur bzw. ihres ähnlichen Verhaltens zu identifizieren sind [WW10, APW08, DDGB09, DDM08]. Oder es wird bereits während der Modellierung eines Prozesses vorgesehen, dass verschiedene Konfigurationen möglich sein sollen [AJ00, HBR08c, ADG$^+$06, LSG09, HBR10a]. Es ist festzustellen, dass in den existierenden Ansätzen für das Management von Prozessvarianten jeweils der Kontrollfluss im Zentrum der Betrachtungen steht, dem Datenfluss wird dagegen kaum Aufmerksamkeit geschenkt. Es ist jedoch aus verschiedenen Gründen wichtig, den Datenfluss beim Variantenmanagement nicht zu vernachlässigen. Einerseits kann sich eine Änderung des Kontrollflusses auch auf den Datenfluss einer Prozessvariante auswirken und andererseits existieren Prozessvarianten, die sich lediglich durch den Datenfluss oder den Datenfluss und die beteiligten Rollen voneinander unterscheiden. Solche *Datenflussvarianten* können beispielsweise entstehen, wenn zwei Unternehmen auf einen Standardprozess zurückgreifen. Während der Kontrollfluss in beiden Firmen gleich ist, gibt es jedoch unterschiedliche Vorgehensweisen, was die Verwaltung der Dokumente betrifft. Während das erste Unternehmen jedes Dokument und jede Information als atomares Element versteht, bündelt das zweite Unternehmen alle zu einem Prozess gehörigen Informationen in einem Dossier. Dies hat zur Folge, dass der Datenfluss der ersten Prozessvariante aus mehreren Datenobjekten besteht, die jeweils individuell durch den Prozess gelenkt werden. Der Datenfluss der zweiten Prozessvariante besteht aus einem einzigen Datenobjekt, dass jeder Aktivität zugänglich gemacht werden muss, die Informationen benötigt.
Das vorliegende Dokument ist wie folgt aufgebaut: In Kapitel 2 werden grundlegende Begriffe eingeführt. Anschließend werden in Kapitel 3 bestehende Forschungsansätze zur Unterstützung von Prozessvarianten beleuchtet. Kapitel 4 widmet sich den Datenflussvarianten und zeigt auf, unter welchen Bedingungen sie entstehen können. Daraufhin werden in Kapitel 5 fachliche Anforderungen und Herausforderungen hinsichtlich der Unterstützung von Datenflussvarianten erläutert. Abschließend werden in Kapitel 6 und 7 die gewonnen Erkenntnisse diskutiert und zusammengefasst sowie ein Ausblick auf weitere Forschungsschwerpunkte gegeben.

2 Grundlagen

2.1 Lebenszyklus von Prozessen

Der Lebenszyklus eines Prozesses kann durch vier Phasen beschrieben werden: *Modellierung*, *Konfiguration*, *Ausführung* und *Analyse & Evaluation*. Die Ergebnisse der letztgenannten Phase gehen wieder in die Modellierungsphase über (vgl. Abbildung 1). In der ersten Phase eines Prozesslebenszyklus wird der Prozess modelliert und das resultierende Modell validiert und verifiziert. Anschließend wird der Prozess implementiert und an die Gegebenheiten des Anwendungskontexts angepasst (z.B. an eine konkrete Organisationsstruktur oder an gesetzliche Bestimmungen). In der darauf folgenden Phase wird der Prozess ausgeführt, d.h. Instanzen des Prozessmodells werden erzeugt und entsprechend der unterschiedlich konfigurierten Prozesslogik ausgeführt. Die letzte Phase eines Zyklus betrifft die Analyse und die Evaluation ausgeführter Prozessinstanzen. In dieser wird der Ablauf der durchgeführten Instanzen analysiert. Auf der Basis der Ergebnisse dieser Phase kann das Prozessmodell angepasst werden, um den jeweils aktuellen Bedürfnissen und Gegebenheiten besser gerecht zu werden [WSR09].

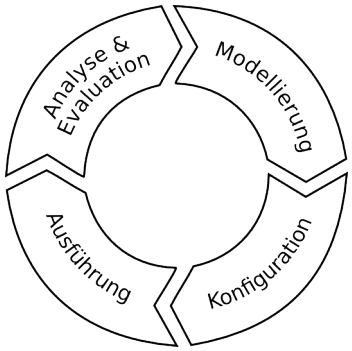


Abbildung 1: Prozesslebenszyklus [Wes07]

2.2 Anpassung von Prozessen

In den Phasen Modellierung, Konfiguration und Ausführung können Anpassungen an einem Prozess bzw. einer laufenden Prozessinstanz vorgenommen werden. Werden in der Modellierungsphase verschiedene Anpassungen an einen Prozess vorgenommen, so dass daraus verschiedene Prozessdefinitionen entstehen, oder werden verschiedene Prozesse zur selben Zielerreichung modelliert, spricht man von *Varianten eines Prozesses* [HBR08b]. In der Konfigurationsphase wird die konkrete Prozessvariante konfiguriert, die in einem bestimmten Anwendungskontext ausgeführt werden soll. Wird eine Prozessinstanz während ihrer Ausführung modifiziert, etwa um einer nicht vorhersehbaren Gegebenheit gerecht zu werden, liegt eine *Ad-hoc-Abweichung bzw. Änderung* vor [RRMD09]. Eine *Version* eines Prozesses entsteht, wenn in der Analyse & Evaluationsphase des Prozesses deutlich wird, dass grundlegende Änderungen an dem Prozess vorgenommen werden müssen und der neue Prozess den alten ersetzt. Durch die Evaluation kann jedoch auch deutlich werden, dass eine neue Variante eines Prozesses benötigt wird. Dies löst die vorhandenen Prozessdefinition jedoch nicht ab, sondern existiert parallel dazu.

2.2.1 Prozessvarianten

Varianten eines Prozesses können auf zwei Arten entstehen. Einerseits kann ein Prozess modifiziert werden, um bestimmten Gegebenheiten besser gerecht zu werden. Dadurch liegen anschließend der ursprüngliche Prozess und speziell ausgerichtete Varianten des Prozesses vor. Je nach Gegebenheit wird ausgewählt, ob der ursprüngliche Prozess oder eine Variante des Prozesses verwendet bzw. ausgeführt werden soll. Andererseits ist es in großen Unternehmen möglich, dass unabhängig voneinander Prozesse zur selben Zielerreichung modelliert werden [WRMR11]. Beispielsweise können an jedem Standort eines Unternehmens Prozesse erstellt werden, die auf die dortigen Bedürfnisse zugeschnitten werden, ohne darauf Bezug zu nehmen, welche Prozesse an anderen Standorten für dasselbe Ziel existieren [GWJV+09]. Generell lässt sich festhalten, dass sich Varianten dadurch auszeichnen, dass sie große Gemeinsamkeiten aufweisen, aber in bestimmten Teilen voneinander abweichen. Sie existieren nebeneinander, d.h. zur selben Zeit. Abbildung 2 illustriert dies, indem explizit kenntlich gemacht wird, dass ein Geschäftsprozess durch mehrere Varianten modelliert werden kann.

In [OF06] wird die Gliederung von Prozessen untersucht, die als *Triage* bzw. *Segmentierung* eines Prozesses bezeichnet wird. Es werden drei mögliche Dimensionen beschrieben: *Funktionen*, *Komplexität* und *Kundengruppen* (vgl. Abbildung 3). Die funktionale Segmentierung wird typischerweise dann verwendet, wenn Spezialkenntnisse für einzelne Aufgaben benötigt werden. Diese Segmentier-

Abbildung 2: Varianten eines Prozesses

ungsdimension entspricht der Gliederung eines Prozesses in Teilprozesse und soll deshalb bei der folgenden Betrachtung von Prozessvarianten vernachlässigt werden. Anders verhält es sich mit der Segmentierung nach Komplexität und Kundengruppen. Bei ersterer werden Prozessvarianten zur Verfügung gestellt, die einen Geschäftsprozess in unterschiedlichen Schwierigkeitsgraden unterstützen. So wird etwa eine Prozessvariante für den Routinefall bereitgestellt, eine für komplexe Fälle und eine Variante für die Fälle, die nicht mehr Routine sind, aber auch nicht zu den komplexen Fällen zählen. In die Prozessvarianten werden dabei jeweils Spezialisten einbezogen. Denkbar ist eine solche Segmentierung z.B. bei der Vergabe von Krediten durch eine Bank. Die dritte Segmentierungsdimension richtet sich nach den Kundengruppen. So wird der Prozess zur Kommunikation von wissenschaftlichen Ergebnissen für das eigene Institut, für die wissenschaftliche Gemeinschaft und die breite Öffentlichkeit unterschiedlich ablaufen und auch unterschiedlich detaillierte Informationen beinhalten. Über die drei Segmentierungsdimensionen von [OF06] hinaus sind Rahmenbedingungen, d.h. der Kontext, als weitere mögliche Dimension zu nennen, durch die Prozessvarianten entstehen. Zu den Rahmenbedingungen zählen z.B. die nationalen, rechtlichen Grundlagen, auf die sich ein international tätiges Unternehmen einstellen muss.

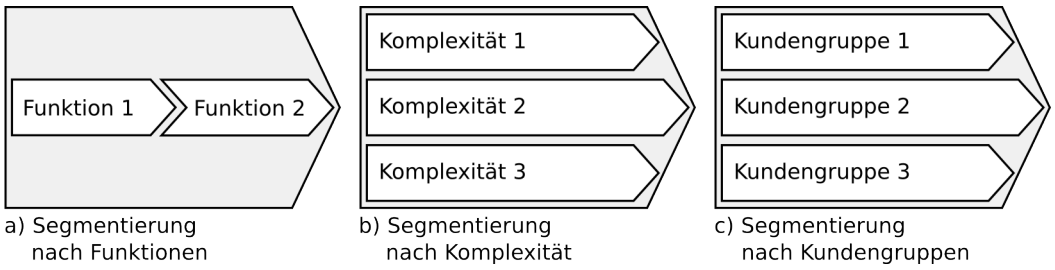


Abbildung 3: Triage bzw. Segmentierung eines Prozesses nach [OF06]

2.2.2 Ad-hoc-Abweichungen

Eine *Ad-hoc-Änderung* ist eine individuelle Abweichung von einem gegebenen Prozessmodell zum Ausführungszeitpunkt, d.h. bezogen auf eine bestimmte Prozessinstanz [RD97, RRMD09, Rei00]. Sie wird auf einer einzelnen Prozessinstanz vorgenommen und ist zum Modellierungszeitpunkt nicht vorhersehbar. In der Evaluationsphase eines Prozesses können alle vorgenommenen Ad-hoc-Änderungen anhand ihrer Ausführungs- und Änderungshistorien betrachtet werden [GRMR$^+$08]. Wird deutlich, dass eine signifikante Anzahl an Ad-hoc-Änderungen ähnlich bzw. gleich sind, deutet dies darauf hin,

dass die Prozessdefinition entweder im Sinne einer neuen Version angepasst, oder eine zusätzliche Prozessvariante geschaffen werden sollte.

2.2.3 Versionen

Eine neue *Version* eines Prozesses kann beispielsweise neue Funktionalitäten aufweisen oder Änderungen der Gesetzeslage berücksichtigen, d.h. sie zeichnet sich gegenüber der bestehenden Version dadurch aus, dass sie grundlegende Änderungen aufweist. In der Übergangszeit werden oft mehrere Versionen verwendet. Prinzipiell löst eine neue Version jedoch die frühere ab (siehe Abbildung 4).

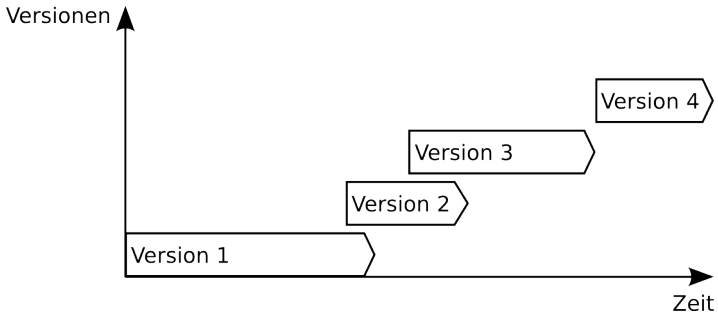


Abbildung 4: Versionen eines Prozesses

2.3 Daten in Prozessmanagementsystemen

2.3.1 Workflow Perspektiven

Wie bereits geschildert, besteht ein Geschäftsprozess aus verschiedenen integralen Bestandteilen, die bei seiner Modellierung berücksichtigt werden müssen. Wenn diese Bestandteile nicht nur gemeinsam, sondern auch voneinander getrennt betrachtet werden können, ist der Prozess besser überschaubar, d.h. leichter zu verstehen, und die Modellierungskomplexität wird verringert. Die Betrachtung von Prozessbestandteilen wird in [CKO92] als Perspektiven der Prozessdarstellung bezeichnet. Es werden die *Funktions-*, *Verhaltens-*, *Organisations-* und die *Informationsperspektive* unterschieden. Die Kombination dieser vier Perspektiven wird als konsistentes und vollständiges Modell eines Prozesses verstanden. Die *Funktionsperspektive* fokussiert darauf, was in einem Prozess getan werden muss, d.h. welche Aktivitäten, während eines Prozesses ausgeführt werden. Wie und wann die Aktivitäten ausgeführt werden, bzw. allgemeiner ausgedrückt der Kontrollfluss eines Prozesses, wird durch die *Verhaltensperspektive* beschrieben. Wer eine Aktivität ausführt, wird durch die *Organisationsperspektive* festgelegt. In ihr können neben Rollen auch Zuständigkeiten, Zugriffsrechte und Verfügbarkeiten definiert werden. Die letzte Perspektive, die *Informationsperspektive*, gibt wieder, welche Daten bzw. Produkte durch den Prozess generiert werden, wie sie während des Prozesses verändert werden, wie sie strukturiert sind und wie sie miteinander zusammenhängen (Datenfluss). Die Idee der Workflow Perspektiven wurde in verschiedenen Forschungsansätzen [AWW03, AJ00] aufgenommen und leicht angepasst. So fasst der in [AJ00] beschriebene Ansatz die Funktions- und die Verhaltensperspektive in der *Prozessperspektive* zusammen. Das bedeutet, dass die Prozessperspektive festlegt, welche Aktivitäten wann und wie ausgeführt werden. Daneben wird die *Operationsperspektive* neu eingeführt. Diese beschreibt die elementaren Operationen, die von den Ressourcen und Anwendungen eines Prozesses ausgeführt werden. Zu diesen Operationen zählen typischerweise das Erstellen, Ändern und Löschen von Daten der Informationsperspektive durch die Aktivitäten der Prozessperspektive. Als

fünfte Perspektive wird die *integrative Perspektive*, eine alles umfassende Perspektive, explizit hinzugefügt.
[AWW03] kennt ebenfalls fünf Perspektiven: Funktionsperspektive, Prozessperspektive, Organisationsperspektive, Informationsperspektive und Operationsperspektive. Verglichen mit der Kategorisierung von [CKO92] unterscheidet sie sich darin, dass die Verhaltensperspektive als Prozessperspektive bezeichnet wird. Zusätzlich wurde die Operationsperspektive aufgenommen, die wie in [AJ00] beschrieben, die elementaren Operationen, die von Ressourcen und Applikationen ausgeführt werden, definiert. Die Gesamtsicht auf alle Perspektiven wird als *Workflow Schema* bezeichnet.
Im Weiteren soll die Perspektivendefinition von [AWW03] mit einer leichten Anpassung und einer Konkretisierung übernommen werden (siehe Abbildung 5). Erstere besteht darin, dass die Prozessperspektive wie bei [CKO92] als Verhaltensperspektive bezeichnet wird. Diese Bezeichnung wird bevorzugt, da der Begriff Prozess in der Regel den gesamten Prozess mit Daten und Akteuren Bezug nimmt, und sich nicht auf die Anordnung der Aktivitäten beschränkt. Zum Zweiten wird die in [AWW03, AJ00] gegebene Definition der Operationsperspektive so interpretiert, dass diese Perspektive darüber Auskunft gibt, wie (erstellen, ändern, löschen) und durch wen (welche Aktivität) bzw. wann (Reihenfolge in der die Aktivitäten Änderungen vornehmen) Datenobjekte verändert werden, d.h. die Operationsperspektive beschreibt den Datenfluss. Die fünf Perspektiven eignen sich gut für die weiteren Betrachtungen, da sie nicht nur zwischen Aktivitäten (Funktionsperspektive), Daten (Informationsperspektive) und beteiligten Rollen (Organisationsperspektive) unterscheiden, sondern auch Kontroll- (Verhaltensperspektive) und Datenfluss (Operationsperspektive) gesondert darstellen. Die Gesamtsicht auf den Prozess ist mehr als eine weitere Perspektive und soll deshalb, wie in [AWW03], als Workflow Schema bezeichnet werden.

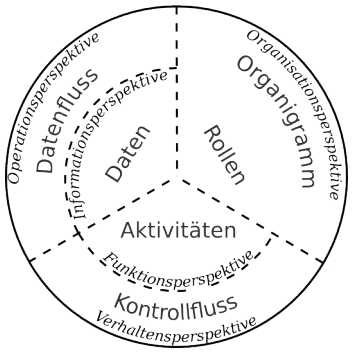


Abbildung 5: Workflow Perspektiven

2.3.2 Daten

Datenkategorisierung Daten dienen dazu, Informationen zu repräsentieren. Aus den Spezifikationen der Workflow Management Coalition [Hol95, Wor99] lässt sich entnehmen, dass Daten zu verschiedenen Zwecken verwendet werden: zum Ersten zur Definition des Prozesses, der Rollen bzw. der Organisationsstruktur sowie der Arbeitsliste, zum Zweiten zum Steuern des Prozesses und zur Verarbeitung, zur Weitergabe und zum Abspeichern von Informationen und schließlich zum Dokumentieren der Ausführungshistorie der Prozessinstanzen. Wird von Daten eines Workflows gesprochen, z.B. in der Informationsperspektive, so ist die zweite Art von Daten gemeint. Die Workflow Management Coalition [Hol95, Wor99] teilt diese Daten wiederum in drei Kategorien ein: *Workflow-Steuerungsdaten*, *Workflow-relevante Daten* und *Anwendungsdaten*. Die beiden erstgenannten wer-

den zur Laufzeit einer Prozessinstanz initialisiert, verwendet und nach deren Ausführung wieder freigegeben, während die Anwendungsdaten unabhängig von einer Prozessinstanz existieren. Wie in Abbildung 6 zu sehen, sind Workflow-Steuerungsdaten interne Daten eines Workflow-Management-Systems. Sie dienen beispielsweise zur Identifikation des Zustands einzelner Aktivitäten oder Instanzen. Workflow-relevante Daten, auch Instanzdaten genannt, dienen zum Feststellen der Zustandsübergänge einer Instanz. Durch sie kann beispielsweise bei bedingten Verzweigungen entschieden werden, welchen Pfad der Kontrollfluss einschlagen muss. Routing-Bedingungen können dabei von Daten abhängen, die durch den Prozess bzw. Anwendungen, die in den Prozess eingebunden sind, bearbeitet werden. Aus diesem Grund können Workflow-relevante Daten durch die verknüpften Anwendungen geändert werden. Persistente Daten, die infolge eines Geschäftsprozesses erstellt, geändert oder gelöscht werden, werden als Anwendungsdaten bezeichnet. Diese Daten sind für das Workflow-Management-System nicht direkt zugänglich, d.h. sie können nur durch verknüpfte Anwendungen, aber nicht durch das Workflow-Management-System selbst manipuliert werden. Das Workflow-Management-System stellt jedoch den Austausch von Daten zwischen mehreren Anwendungen, der gegebenenfalls mit einer Datentypumwandlung verbunden ist, sicher. Dieser Austausch wird mithilfe von Workflow-relevanten Daten vorgenommen.

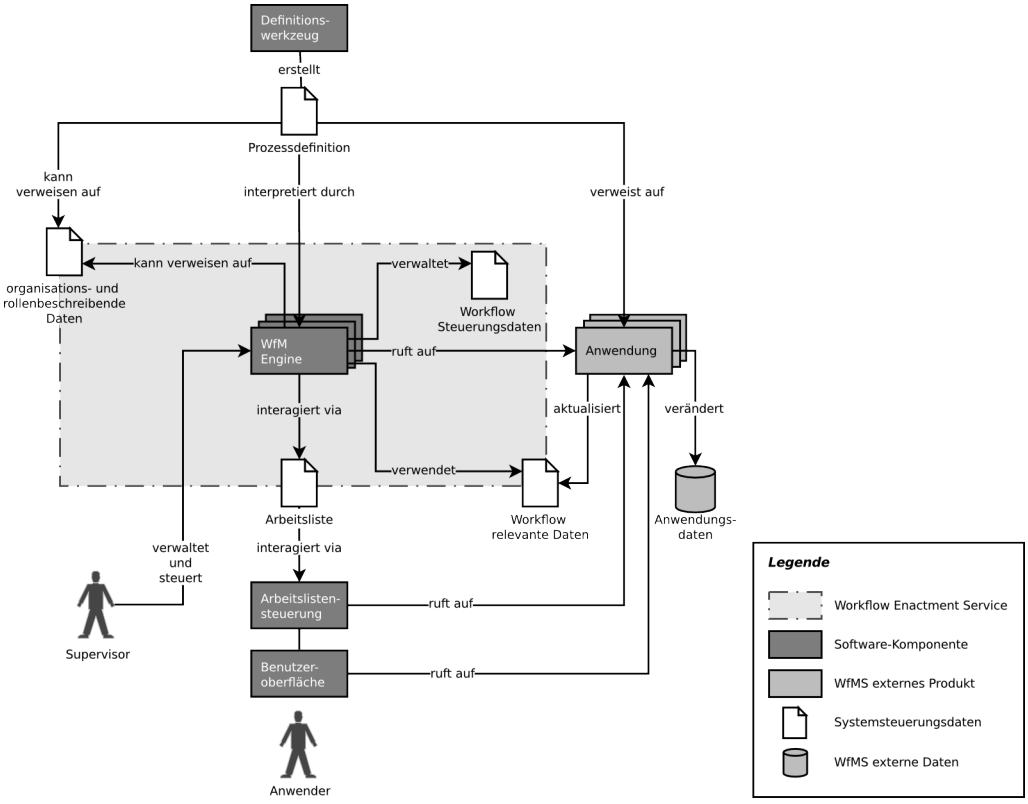


Abbildung 6: Generische Struktur eines Workflow-Management-Systems [Hol95]

Eine weitere Datenkategorisierung wird in [AWW03] vorgenommen. Dabei wird zwischen zwei Datenkategorien unterschieden: *Steuerungsdaten* und *produktive Daten*. Steuerungsdaten sind die Daten, die für interne Zwecke des Workflow-Management-Systems verwendet werden, wobei Informationen für das datenbasierte Routing eingeschlossen sind. Produktive Daten werden hingegen als die Daten beschrieben, die unabhängig vom Workflow-Management-System existieren.

Im Vergleich zu [Hol95] ist dieser Ansatz eine vereinfachte Kategorisierung, da eine klare Trennung zwischen den Datenarten vorgenommen wird, ohne darauf hinzuweisen, dass die Daten für das datenbasierte Routing in der Schnittmenge von Steuerungsdaten und produktiven Daten liegen.

In [SOSF04] wird eine Unterscheidung von vier Kategorien vorgenommen: *Referenzdaten*, *operative Daten*, *Entscheidungsdaten* und *Kontextdaten*. Diese Einteilung deckt sich in weiten Teilen mit der Kategorisierung der Workflow Management Coalition, fokussiert aber stärker auf die Sicht des Anwenders als auf die Sicht des Workflow-Management-Systems. Referenzdaten sind Systemdaten, die dazu dienen die Instanzen eines Prozesses zu identifizieren. Als operative Daten werden die Daten bezeichnet, die von den Aktivitäten eines Prozesses gelesen, weiterverarbeitet und geschrieben werden. Entscheidungsdaten sind für das datenbasierte Routing essentiell. Sie müssen den entsprechenden Entscheidungsknoten im Prozess zur Verfügung stehen. Kontextdaten sind die Eingabe oder das Ergebnis eines Prozesses. Typischerweise liegen Kontextdaten in einer komplexen Datenstruktur vor. Ein Beispiel für Kontextdaten, die das Ergebnis eines Prozesses sind, sind eine Immatrikulationsbescheinigungen, die nach erfolgreicher Bewerbung an einer Hochschule ausgestellt werden.
Bezogen auf die Kategorien von [Hol95] lässt sich feststellen, dass Referenzdaten eine Teilmenge der Workflow-Steuerungsdaten sind. Sie können unter bestimmten Umständen auch Teil der Anwendungsdaten sein, beispielsweise dann, wenn eine Auftragsnummer (Anwendungsdatum) zur internen Referenzierung (Workflow-Steuerungsdatum) einer Instanz verwendet wird. Des Weiteren stellen Entscheidungsdaten sowohl eine Teilmenge der Workflow-Steuerungsdaten als auch der Anwendungsdaten dar. Der Austausch zwischen Workflow-Steuerungsdaten und Anwendungsdaten wird jeweils über die Workflow-relevanten Daten vorgenommen. Schließlich entsprechen die operativen Daten, d.h. insbesondere die Kontextdaten, den Anwendungsdaten.

Für die weiteren Betrachtungen werden drei Hauptdatenkategorien unterschieden (siehe Abbildung 7): Steuerungsdaten, operative Daten und die Schnittmenge beider Kategorien.

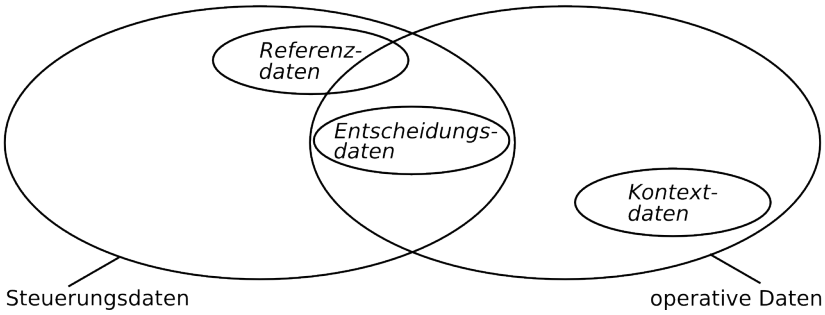


Abbildung 7: Datenkategorien und relevante Unterkategorien

Steuerungsdaten. Hierbei handelt es sich um Workflow-Management-System interne Daten, die dazu dienen den Prozess bzw. die erstellen Prozessinstanzen zu verwalten. Im Weiteren ist jedoch nur eine Teilmenge dieser Daten relevant: *Referenzdaten*. Sie können unter anderem dazu dienen, operative Daten verschiedener Instanzen voneinander zu unterscheiden. Referenzdaten können sich auf die gesamte Prozessinstanz beziehen oder auf eine einzelne Aktivität, die mehrfach instanziiert wird. In [PDKL09] wird die mehrfache Instanziierung einer Aktivität anhand des Prozesses zur Buchbestellung verdeutlicht. Nachdem eine Kundin bzw. ein Kunde ein bestimmtes Buch bestellt hat, fragt die Buchhandlung bei verschiedenen Lieferanten nach, ob das Buch lieferbar ist und zu welchem Preis. Diese Aktivität wird für jeden Lieferanten einmal, d.h. aus Sicht des Prozesses mehrfach ausgeführt.

Genauso wird von jedem Lieferanten eine Rückmeldung erhalten. [RR06] zeigt, wie bei solchen Multiinstanzaktivitäten die Prozesssteuerung und Ausnahmebehandlung datengetrieben erfolgen kann.
Abbildung 8 zeigt die Referenzdaten einer mehrfach instanziierten Aktivität. Diese setzen sich aus zwei Datenobjekten zusammen: einem Datum, das die jeweils vorliegende Instanz einer Aktivität identifiziert und einer vollständigen Liste aller Instanzen einer Aktivität. Referenzdaten sind für Datenflussvarianten wichtig, da operative Daten potentiell mit einer mehrfach instanziierten Aktivität verbunden sind, d.h. eine Aktivität nicht nur mit einem Datenobjekt verknüpft ist, sondern ein Datenobjekt je Aktivitätsinstanz existiert. Referenzdaten, die Prozessinstanzen voneinander unterscheiden, sind für Datenflussvarianten nicht relevant.

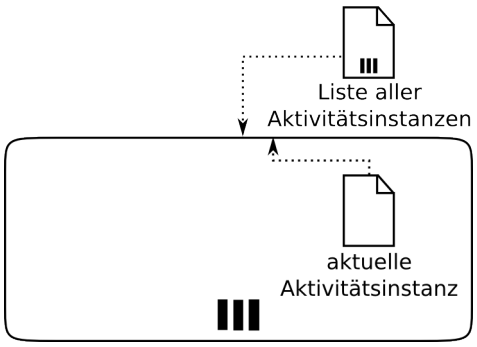


Abbildung 8: Darstellung von Referenzdaten bei Mehrfachinstanziierung einer Aktivität

Operative Daten. Diese Daten werden von den Aktivitäten eines Prozesses als Eingabe benötigt bzw. als Ergebnis generiert. Wie in Abbildung 9a zu sehen, werden operative Daten explizit durch Datenobjekte dargestellt. Ein Datenfluss entsteht, indem eine Aktivität A ein Datenobjekt D mit dem Wert x liest, den Wert x' schreibt und es einer nachfolgenden Aktivität zur Verfügung stellt. *Kontextdaten*, eine Teilmenge der operativen Daten, zeichnen sich dadurch aus, dass sie entweder Ein- oder Ausgabe einer ganzen Prozessinstanz sind und nicht selbst durch eine Prozessinstanz fließen. Entweder initialisieren sie eines oder mehrere Datenobjekte oder sie fassen mehrere durch die Prozessinstanz erzeugte Datenobjekte zusammen. Typischerweise weisen Kontextdaten eine komplexe Datenstruktur auf. Abbildung 9b zeigt, wie einer Aktivität A ein Kontextdatenobjekt K mit dem Wert xyz lesend zur Verfügung gestellt wird. Diese Information wird verwendet, um das Datenobjekt D mit dem Wert y zu initialisieren und mit diesem Datenobjekt weiterzuarbeiten. Ein Beispiel für Eingabekontextdaten sind Antragsformulare, aus denen verschiedene Informationen ausgelesen werden und anhand dieser Informationen weitere Schritte eingeleitet werden. Abbildung 9c verdeutlicht, wie Aktivität A auf die Datenobjekte D_1 mit dem Wert x und D_2 mit dem Wert z lesend zugreift und das Kontextdatenobjekt K im Wert xz ausgibt. Beispielsweise kann ein solches Ausgabekontextdatenobjekt ein Zwischenzeugnis mit vorläufigen Notendurchschnitt sein. Eingabekontextdaten (Speisung eines Prozesses) können nur gelesen werden, während Ausgabekontextdaten (Produkt eines Prozesses) nur erstellt werden können, andere Operationen sind nicht zulässig. Dies wird in Abbildung 9 dadurch deutlich, dass Eingabekontextdaten nur ausgehende Datenflusskanten besitzen, während Ausgabekontextdaten nur eingehende Kanten aufweisen können.

Operative Daten lassen sich in interne und externe operative Daten unterscheiden. Interne operative Daten werden zur Ausführung einer Prozessinstanz erstellt und während ihrer Lebensdauer verwendet. Beim Beenden der Instanz werden sie gelöscht. Externe operative Daten existieren hingegen über die Ausführungsdauer einer Prozessinstanz hinaus, etwa eine Datenbank, in der Studierendendaten gespeichert sind.

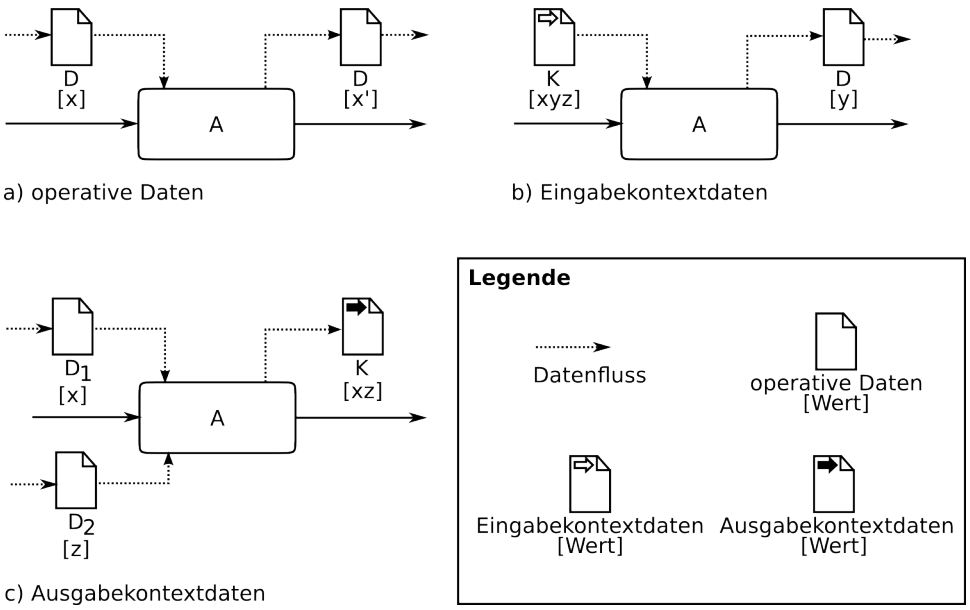


Abbildung 9: Darstellung von operativen Daten

Schnittmenge von Steuerungsdaten und operativen Daten. In der Schnittmenge von Steuerungsdaten und operativen Daten liegen sowohl eine Teilmenge der Referenzdaten, als auch die vollständige Menge der Entscheidungsdaten.
Referenzdaten wurden bereits eingehend als Teilmenge der Steuerungsdaten beschrieben. Sie dienen dazu, verschiedene Instanzen voneinander zu unterscheiden. Geschieht dies durch systeminterne Identifikatoren, so handelt es sich um reine Steuerungsdaten. Beziehen sich die Referenzdaten auf operative Daten, um eine Aktivitätsinstanz zu identifizieren, so liegen sie in der Schnittmenge von Steuerungsdaten und operativen Daten. Etwa kann eine Auftragsnummer zur Referenzierung benutzt werden.
Entscheidungsdaten gehören sowohl zu den Steuerungsdaten, als auch zu den operativen Daten. Sie unterstützen das datenbasierte Routing aufgrund von Daten, die durch die Aktivitäten des Prozesses bearbeitet werden. Entscheidungsdaten sind sowohl ein Bindeglied zwischen systeminternen und -externen Daten, als auch Bindeglied zwischen Kontroll- und Datenfluss. Um dies in einer graphischen Prozessdarstellung zu berücksichtigen, sollen (abweichend von der Standarddarstellung) bedingte Verzweigungsknoten (`OR`- und `XOR`-Gateways) mit den entsprechenden operativen Datenobjekten explizit lesend verbunden werden (Abbildung 10). Gateways dürfen jedoch keine Daten in den Datenfluss speisen, d.h. es darf keine schreibende Kante von einem Verzweigungsknoten zu einem Datenobjekt ausgehen.

Datenstruktur operativer Daten Die Workflow Management Coalition [Wor99] hält zu den Workflow-relevanten Daten fest, dass sie typisiert oder untypisiert sein können. Während erstere durch das Workflow-Management-System verarbeitet werden können, kann es untypisierte Daten nicht interpretieren und lediglich an Applikationen weitergeben. Diese Sicht ist rein auf die Funktionalität des Workflow-Management-Systems bezogen. Sie thematisiert nicht, wie sichergestellt wird, dass die weitergegebenen Daten von der Applikation, die sie erhält, interpretiert und verarbeitet werden können. Gerade dies muss jedoch aus Sicht des Gesamtprozesses zwingend sichergestellt werden.
Daten können prinzipiell atomar oder komplex sein. Komplexe Daten können wiederum strukturiert,

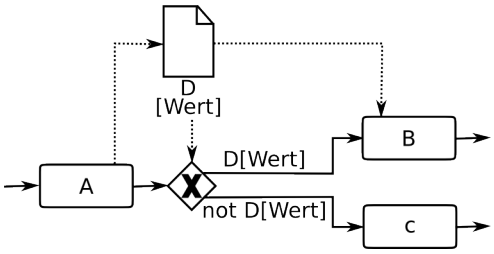


Abbildung 10: Darstellung von Entscheidungsdaten

semistrukturiert oder unstrukturiert vorliegen. Um die Daten verarbeiten zu können, muss die Datenstruktur festgelegt werden, d.h. für jede Aktivität, die Daten als Eingabe erwartet, muss definiert werden, in welcher Form diese vorliegen sollen. Ebenso müssen die Ergebnisdaten einer Aktivität spezifiziert werden. Dies ist besonders zu beachten, wenn ein Geschäftsprozess in mehreren Systemen oder zwischen mehreren Akteuren abläuft. Bei der Modellierung eines Prozesses ist deshalb die Datenstruktur jedes Datenobjekts festzulegen.
Für die weiteren Betrachtungen wird festgehalten, dass Datenobjekte mit demselben Namen, die von Akteuren derselben Organisationszugehörigkeit (in der Modellierungsnotation für Geschäftsprozesse (BPMN) [OMG11] innerhalb derselben Swimlane bzw. demselben Pool) verwendet werden, dieselbe Datenstruktur aufweisen oder implizit in die benötigte Datenstruktur umgewandelt werden. Es ist zu beachten, dass keine Aussage über die Datenstruktur getroffen wird, wenn Akteure unterschiedlicher Organisationszugehörigkeit ein Datenobjekt mit demselben Namen bearbeiten. Auch dann nicht, wenn das Datenobjekt zwischen ihnen ausgetauscht wird.

Charakteristika operativer Daten [RHEA04] beschreibt vier Charakteristika von Daten in Workflows: *Visibilität*, *Interaktion*, *Transfer* und *datenbasiertes Routing*. Dabei bezieht sich die Visibilität auf die operativen Daten, während sich die übrigen Eigenschaften auf den Datenfluss oder den Daten- und Kontrollfluss beziehen.
Die Visibilität legt fest, welche Aktivitäten auf die Daten zugreifen können. Beispielsweise kann ein Wert nur für eine Aktivität, für alle Instanzen dieser Aktivität, für alle Aktivitäten einer Prozessinstanz oder für alle Aktivitäten aller Prozessinstanzen zugänglich sein. Operative Daten sind in der Regel für eine Instanz relevant. Innerhalb dieser Instanz sind sie für die Aktivitäten sichtbar, mit denen sie über eine eingehende Datenkante verbunden sind. Kontextdaten, eine Teilmenge der operativen Daten, können eine Ausnahme bilden und weiterreichende Sichtbarkeit besitzen. Sie können für eine Instanz, den gesamten Prozess, d.h. alle Instanzen des Prozesses, oder für mehrere Prozesse relevant sein. Als Beispiel kann wiederum die Immatrikulation an einer Hochschule dienen. Der Antrag, der von einem Bewerber bzw. einer Bewerberin gestellt wird, ist für eine Instanz relevant. Die Vorlage bzw. das Muster des Antrags ist für den gesamten Prozess von Belang, d.h. sie bzw. es muss in jeder Prozessinstanz zur Verfügung stehen. Die Stammdaten der Person sind nach der erfolgreichen Immatrikulation wiederum für andere Prozesse, wie z.B. für die Prüfungsanmeldung, relevant.

Schnittstellen von Informationsperspektive zu den übrigen Perspektiven Ein Workflow Schema besteht wie in Abbildung 5 zu sehen, aus mehreren Perspektiven. Sie sollen es erleichtern, einen Prozess zu modellieren. Zwischen den Perspektiven auf Daten, Aktivitäten und Rollen, einschließlich deren jeweiliger Anordnung, bestehen sich beeinflussende Beziehungen.
Daten und Datenfluss sowie Aktivitäten und Kontrollfluss stehen zueinander in enger Beziehung.

Einerseits setzt der Daten- bzw. Kontrollfluss voraus, dass die entsprechenden Datenobjekte bzw. Aktivitäten definiert sind. Andererseits können die Daten nur durch einen festgelegten Datenfluss gezielt manipuliert werden, und Aktivitäten können nur ausgeführt werden, wenn sie Bestandteil eines Kontrollflusses sind. Daten- und Kontrollfluss können über das sogenannten datenbasierten Routing gegenseitig aufeinander Auswirkung haben. Daten und Rollen sind miteinander über Zugriffsrechte und Vertraulichkeitsstufen verbunden. Die Verknüpfung von Aktivitäten und Rollen wird durch Zuständigkeiten und Verfügbarkeiten bestimmt. Für die Betrachtungen zu Datenflussvarianten sind die Schnittstellen zwischen den Datenobjekten bzw. dem Datenfluss und den Aktivitäten, dem Kontrollfluss sowie den Rollen relevant.
Um das datenbasierte Routing zu ermöglichen, müssen den entsprechenden Gateways die operativen Daten zur Verfügung stehen. Bei der Modellierung wird dies sichtbar gemacht, indem die entsprechenden Datenobjekte den Verzweigungsknoten explizit über eine Verbindungskante lesend zur Verfügung gestellt werden.
Wie bereits erwähnt, verknüpfen Zugriffsrechte bzw. Vertraulichkeitsstufen die Daten mit den Rollen eines Prozesses. Sie legen fest, welche Rollen auf welche Daten zugreifen dürfen, z.B. bei sensiblen, personenbezogenen Daten (beispielsweise Kundendaten bei Banken). Verschiedene Akteure, wie der Bankmanager und der Schalterangestellte, haben unterschiedliche Rechte, Informationen eines Kunden während des Prozesses „Kreditvergabe“ zu sehen.

2.3.3 Datenfluss

Der Datenfluss eines Prozesses beschreibt, welche operativen Daten welchen Aktivitäten in welcher Reihenfolge zur Verfügung stehen müssen. Dies geschieht, indem bei der Modellierung eines Prozesses für jede Aktivität festgehalten wird, welche Daten sie als Eingabe benötigt und welche Daten als Ausgabe generiert werden. Zudem wird definiert, welche Aktivität diese Daten zuvor als Ausgabe geschrieben hat, d.h. von welcher Aktivität der Wert der erhaltenen Daten abhängt. Gibt es keine andere Aktivität, welche die Daten bereits geschrieben hat, so werden die Daten von dieser Aktivität initialisiert. Auf Datenobjekten können die Operationen `create`, `read`, `write`, `update`, `delete` und `destroy` ausgeführt werden. Die Operationen, die den Aktivitäten zur Verfügung stehen, können auf `read` und `write` reduziert werden [SZNS06]. Wird ein Datenobjekt zum ersten Mal durch eine Aktivität in einem Prozess geschrieben, so wird es gleichzeitig erstellt. Wird der Wert eines Datenobjekts geändert, so wird es geschrieben. Wird eine Instanz beendet, so werden die internen operativen Daten gelöscht.

Darstellung Der Datenfluss eines Prozesses kann auf verschiedene Weise dargestellt werden. Zur graphischen Darstellung können Diagramme verwendet werden. In diesen werden Aktivitäten und Datenobjekte mit Datenflusskanten verbunden (vgl. Abbildung 9). Eine von einem Datenobjekt in eine Aktivität eingehende Kante bedeutet lesenden Zugriff, ausgehende Kanten bedeuten, dass der Wert des Datenobjekts schreibend verändert werden kann. Um den Datenfluss eines Datenobjekts sichtbar zu machen, trägt das von einer Aktivität weitergegebene Datenobjekt denselben Namen, wie das gelesene. Eine Ausnahme bilden Kontextdaten, die nicht durch den Prozess fließen. Sie dienen entweder dazu, ein operatives Datenobjekt zu initialisieren, oder ein komplexe Ausgabe aus einem oder mehreren Datenobjekten zu erzeugen. Dargestellt werden operative Daten durch ein Dokumentensymbol. Der Name des Datenobjekts wird direkt unter diesem angegeben. Darunter folgt wiederum der Wert des Datenobjekts.
Um den Datenfluss aller Datenobjekte formal darzustellen, kann eine *Datenflussmatrix* verwendet werden. In [SZNS06] wird eine Datenflussmatrix definiert, deren Spalten die Aktivitäten und deren

Zeilen die atomaren Datenobjekte beschreibt. Die Elemente der Matrix sind die Operationen lesen und schreiben. Die Matrix enthält dabei genau dann einen Eintrag, wenn eine Aktivität auf einem Datenobjekt eine entsprechende Operation ausführt.

In den folgenden Betrachtungen wird der Datenfluss operativer Daten explizit dargestellt, d.h. Daten- und Kontrollfluss werden getrennt modelliert (vgl. Abbildung 11b). In Anlehnung an BPMN 2.0 [OMG11] sind der Beginn und das Ende eines Datenflusses durch Eingabe- und Ausgabedatenobjekte zu kennzeichnen. Externe Datenquellen werden dargestellt. Sie werden in der Regel zur Initialisierung oder Ablage eines Datenobjektes benutzt. Wie in Abbildung 7 zu sehen, sind Entscheidungsdaten eine Teilmenge der operativen Daten und gleichzeitig eine Teilmenge der Steuerungsdaten. Um dieser Tatsache gerecht zu werden, werden die Entscheidungsknoten im Folgenden explizit lesend mit dem Datenfluss verbunden (vgl. Abbildung 10). Diese Modellierung ist in BPMN 2.0 nicht vorgesehen.
Der Datenfluss der Referenzdaten verläuft entlang des Kontrollflusses, d.h. wird nicht mittels separaten Datenkanälen modelliert.

Implementierungsmodelle Der Datenfluss kann in einem Workflow-Management-System auf unterschiedliche Weise beschrieben werden [SOSF04, RHEA04]. Abbildung 11 zeigt drei denkbare Modelle:

a) *Impliziter Datenfluss.* Kontroll- und Datenfluss sind miteinander verschmolzen. Die Datenobjekte werden allen Aktivitäten entlang des Kontrollflusses zur Verfügung gestellt, unabhängig davon, ob sie diese Daten als Eingabe erfordern (siehe Abbildung 11a).

b) *Expliziter Datenfluss.* Kontroll- und Datenfluss werden separat modelliert. Dabei werden die Daten von einer Aktivität zur anderen weitergegeben und nur die Aktivitäten, für die das Datenobjekt als Eingabe definiert wurde, im Datenfluss berücksichtigt (siehe Abbildung 11b).

c) *No Data Passing / Impliziter Datenfluss über einen Datenspeicher.* Die Aktivitäten rufen die benötigten Eingabedatenobjekte aus einem globalen Datenspeicher ab, in die sie zuvor von anderen Aktivitäten als Ausgabe abgelegt wurden (siehe Abbildung 11c). [RHEA04] spricht in diesem Zusammenhang von „no data passing“, d.h. keiner Weitergabe der Daten von einer Aktivität zur anderen. [SOSF04] beschreibt diese Konstellation als impliziten Datenfluss über einen Datenspeicher. Implizit vermutlich deshalb, weil nur durch die Verfolgung des Kontrollflusses festgestellt werden kann, welche Aktivität das Datenobjekt zuletzt geschrieben hat und jede Aktivität potentiell auf alle abgelegten Daten des Datenspeichers zugreifen kann.

In [RHEA04] werden darüber hinaus Varianten für die Implementierung des Datenaustauschs zwischen einer komplexen Aktivität und dem Teilprozess, den sie repräsentiert, beschrieben. Zudem wird gezeigt, wie Daten, die mit einer Aktivität verbunden sind, die mehrfach instanziiert wird, gehandhabt werden können. Schließlich wird erläutert, wie Daten zwischen verschiedenen Instanzen eines Prozesses ausgetauscht werden können. Während diese Datenweitergabe stets innerhalb eines Prozesses stattfindet, werden weitere Modelle für den Austausch von Daten einer Aktivität, einer Instanz bzw. einem Prozess und Daten, die außerhalb des Workflow-Management-Systems existieren, vorgestellt.

Datentransfer [RHEA04] stellt eine Vielzahl von Transfermechanismen vor. Zu diesen zählen die Übergabe als Wert (*pass by value*), als Referenz (*pass by reference*), als Kopie (*copy in/out*) sowie die Datenweitergabe verbunden mit Datentransformationen. Bei der Datenübergabe in Form von Werten werden diese direkt zwischen den Aktivitäten der Prozessinstanz ausgetauscht, d.h. es wird kein

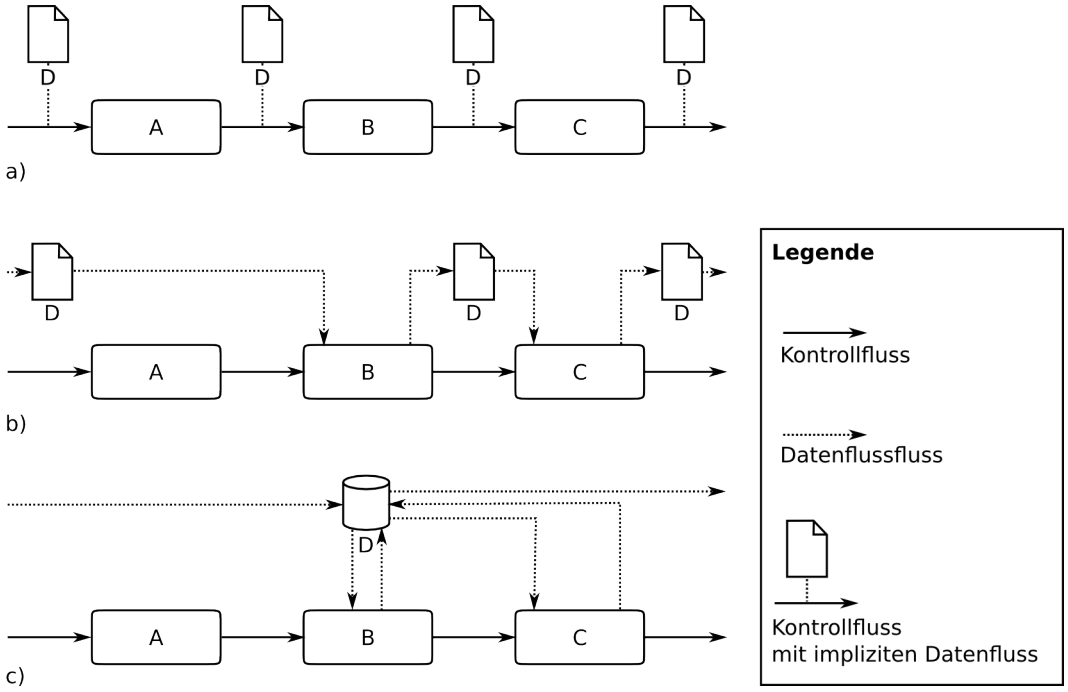


Abbildung 11: Datenflussimplementierungsmodelle [SOSF04] bzw. möglicher Datenaustausch zwischen Aktivitäten [RHEA04]

gemeinsamer Datenspeicher benötigt. Hingegen wird beim zweiten Transfermechanismus auf Datenobjekte in einem gemeinsamen Datenspeicher referenziert, d.h. die zugreifende Aktivität ändert das Datenobjekt direkt im Datenspeicher. Wie bei einer Datenbank ist es bei diesem Mechanismus möglich, dass auf ein Datenobjekt gleichzeitig zugegriffen wird. Aus diesem Grund wird zwischen einem Referenztransfermechanismus mit und einem ohne Zugriffssperre unterschieden. Die Datenübergabe in Form einer Kopie erfolgt indem ein Datenobjekt aus dem Datenspeicher kopiert, von einer Aktivität verarbeitet und anschließend zurückgeschrieben wird. Dies bedeutet insbesondere, dass sobald eine Aktivität ein Datenobjekt kopiert hat, diese nicht mehr darauf lesend zugreift. Eine zwischenzeitliche Änderung durch eine andere Aktivität kann somit nicht berücksichtigt werden. Schließlich ist der Datentransformationsmechanismus zu nennen. Vor der Eingabe in eine Aktivität oder bei der Ausgabe einer Aktivität kann ein Datenobjekt in die nachfolgend benötigte Datenstruktur transformiert werden. Bei der Datenübergabe zwischen Workflow-internen und externen Daten werden zusätzlich Mechanismen unterschieden, welche Daten nachfragen (*pull*) bzw. Daten bereitstellen (*push*). Erstere zeichnen sich dadurch aus, dass sie die benötigten Daten anfordern, während letztere den Datentransfer initiieren, indem sie Daten weitergeben.

3 Stand der Technik

Es wurde schon früh beobachtet, dass Prozesse nicht starr sind, sondern während ihres Lebenszyklus Änderungen im Sinne von Varianten, Ad-hoc-Abweichungen und Versionen erfahren [WSR09]. Varianten können auf unterschiedliche Weise entstehen: sie können gezielt zum Modellierungszeitpunkt des Prozesses vorgesehen werden, d.h. auf einem gemeinsamen Modell basieren, oder sie können unabhängig voneinander modelliert werden. Letzteres ist beispielsweise möglich, wenn jede Fakul-

tät einer Universität einen Prozess zur selben Zielerreichung modelliert, ohne dass diese Prozesse untereinander abgesprochen und abgeglichen werden. Die Prozesse werden in dem universitären Prozessmanagementsystem verwaltet und sind a priori nicht als Varianten zu erkennen.
Die beiden vorgestellten Variantenarten erfordern jeweils spezifische Lösungsansätze, um möglichst effizient verwendet werden zu können. Im ersten Fall ist die Modellierung, das Bestimmen der auszuführenden Variante und die Evolution der Varianten, d.h. der Lebenszyklus der Varianten, im Fokus des Interesses. Im zweiten Fall hingegen ist das Erkennen von Varianten entscheidend, d.h. mittels Ähnlichkeitsanalysen werden die existierenden Prozesse daraufhin untersucht, ob zwei oder mehrere Prozessmodelle zur selben Zielerreichung dienen.
Im Folgenden gehen wir zuerst auf zwei Ansätze zur gezielten Modellierung von Varianten ein. Anschließend zeigen wir die Möglichkeiten zum Bestimmen von Varianten auf.

3.1 Variantenmodellierung durch Transformation

Ausgangspunkt dieses Ansatzes bilden ein Prozessmodell und eine Menge definierter Transformationsregeln, die auf dieses Modell angewandt werden können. Das Prozessmodell, das im Folgenden als *Basisprozess* bezeichnet wird, kann unterschiedlich gestaltet sein: es kann einen ausführbaren Prozess darstellen, beispielsweise die am häufigsten verwendete Prozessdefinition, oder es kann ein minimales Prozessfragment sein, das allen Varianten zu Grunde liegt [HBR10b]. Im Folgenden sollen zwei Vorgehensweisen zur Variantenmodellierung durch Transformation vorgestellt werden: zum einen ein Ansatz, der auf dem objektorientierten Paradigma der Vererbung beruht, zum anderen eine Vorgehensweise, die modulare, in Kombination anwendbare Prozessanpassungen definiert.

Objektorientierte Vorgehensweise Eine Möglichkeit Varianten zu modellieren, ist sie über Vererbungsmechanismen zu generieren. In [AJ00, AB02, Aal03] werden vier Regeln vorgestellt, mit denen aus einem Basisprozess abgeleitete Prozesse erstellt werden können. Ein abgeleiteter Prozess ist ein Prozess, der alle Aktivitäten des Basisprozesses aufweist und darüber hinaus neue zusätzliche Aktivitäten umfassen kann. Das Verhalten der Aktivitäten, die der Basisprozess und der abgeleitete Prozess gemeinsam haben, ist identisch, d.h. in Bezug auf diese Aktivitäten besitzen der Basisprozess und der abgeleitete Prozess dasselbe Verhalten. Es werden folgende Vererbungsarten unterschieden:

- *Protokollvererbung (Protocol Inheritance).* Protokollvererbung zeichnet sich dadurch aus, dass sich das Verhalten von Basisprozess und abgeleitetem Prozess nicht unterscheiden lässt, wenn bei der Ausführung des abgeleiteten Prozesses nur die Aktivitäten berücksichtigt werden, die auch im Basisprozess enthalten sind. Dies impliziert zum einen, dass der Kontrollfluss des Basisprozesses im abgeleiteten Prozess enthalten ist und zum anderen, dass der abgeleitete Prozess ausführbar ist, wenn die gegenüber dem Basisprozess hinzugefügten Aktivitäten blockiert *(Blocking)* werden. Ein Beispiel für diese Vererbungsart ist das Einfügen eines alternativen Ausführungspfads in einen Prozess. Werden im abgeleiteten Prozess die Aktivitäten des hinzugefügten Alternativpfades blockiert, so verhält sich der abgeleitete Prozess wie der Basisprozess.

- *Projektionsvererbung (Projection Inheritance).* Lässt sich das Verhalten von Basisprozess und abgeleitetem Prozess nicht unterscheiden, wenn alle Aktivitäten des abgeleiteten Prozesses ausgeführt werden, aber nur das Verhalten der Aktivitäten betrachtet wird, die auch im Basisprozess vorhanden sind, so handelt es sich um Projektionsvererbung. Bei dieser Vererbungsart ist der Kontrollfluss des Basisprozesses ebenfalls im abgeleiteten Prozess vorhanden, er lässt sich bei

reiner Projektionsvererbung jedoch nur ausführen, wenn die neu eingefügten Aktivitäten ebenfalls ausgeführt werden. Um die Prozesse vergleichen zu können, wird von dem Verhalten der neuen Aktivitäten abstrahiert, d.h. ihr Verhalten wird versteckt *(Hiding)*. Ein Beispiel für Projektionsvererbung ist das Einfügen eines parallelen Ausführungspfads in einen Prozess. Wird im abgeleiteten Prozess vom Verhalten des hinzugefügten parallelen Ausführungspfads abstrahiert, so verhalten sich abgeleiteter Prozess und Basisprozess gleich. Ein weiteres Beispiel für die Projektionsvererbung ist das sequenzielle Einfügen einer Aktivität in einen Prozess.

- *Protokoll-/Projektionsvererbung (Protocol/Projection Inheritance)*. Ein unter Protokoll-/Projektionsvererbung abgeleiteter Prozess entspricht sowohl der Protokollvererbung, als auch gleichermaßen der Projektionsvererbung. Das bedeutet, dass das Verhalten von Basisprozess und abgeleitetem Prozess weder zu unterscheiden ist, wenn die neu eingefügten Aktivitäten nicht ausgeführt werden (*Blocking*, Protokollvererbung), noch wenn sie ausgeführt werden, aber von ihrem Verhalten abstrahiert wird (*Hiding*, Projektionsvererbung). Beide Vererbungsarten bedingen für das Prozessmodell des abgeleiteten Prozesses, dass der Kontrollfluss des Basisprozesses enthalten ist. Darüber hinaus gibt die Protokollvererbung vor, dass sich die neu eingefügten Aktivitäten auf einem Alternativpfad befinden. Dadurch können die Aktivitäten des Basisprozesses ausgeführt werden, auch wenn die neu eingefügten Aktivitäten blockiert werden. Um auch der Projektionsvererbung zu genügen, muss sichergestellt werden, dass ein Ausführungsszenario existiert, bei dem alle Aktivitäten des abgeleiteten Prozesses ausgeführt werden können. Aus diesen Bedingungen resultiert, dass die neuen Aktivitäten nicht alternativ zu einer oder mehreren Aktivitäten des Basisprozesses eingefügt werden, sondern eine Alternative zu einem Kontrollflusspfad des Basisprozesses darstellen.

- *Lebenszyklusvererbung (Life-Cycle Inheritance)*. Ebenfalls eine Kombination von Protokoll- und Projektionsvererbung ist die Lebenszyklusvererbung. Sie liegt vor, wenn sich das Verhalten von abgeleitetem Prozess und Basisprozess nicht unterscheiden lässt, wenn die in den abgeleiteten Prozess eingefügten Aktivitäten blockiert oder versteckt werden. Das bedeutet, jede der oben genannten Vererbungsarten entspricht gleichzeitig der Lebenszyklusvererbung.

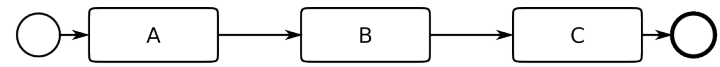
Um einen Prozess aus einem Basisprozess abzuleiten, werden vier Transformationsregeln definiert. Im Folgenden werden diese vorgestellt und durch Abbildung 12 illustriert. Abschließend gibt Tabelle 1 einen Überblick über die vorgestellten Vererbungsarten und die zugehörigen Transformationsregeln. Für detaillierte Informationen verweisen wir auf [AJ00, AB02, Aal03].

- `PT`. Wird die Transformationsregel `PT` auf den Basisprozess angewandt, erhält man einen unter Protokoll- und Lebenszyklusvererbung abgeleiteten Prozess. Diese Regel fügt Aktivitäten auf Alternativpfaden in den Basisprozess ein (siehe Abbildung 12c).

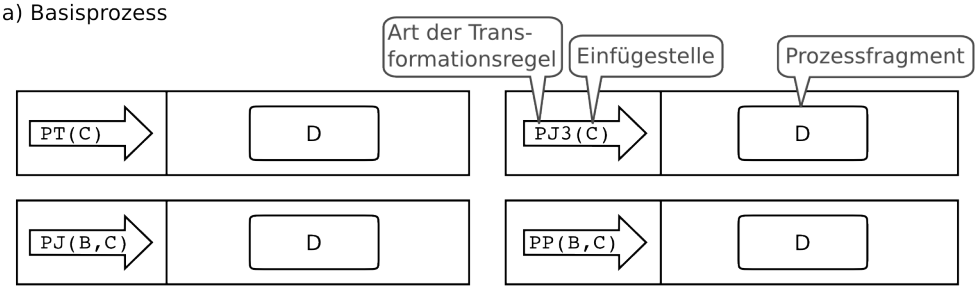
- `PJ`. Diese Transformationsregel fügt Aktivitäten sequenziell in den Kontrollfluss des Basisprozesses ein (siehe Abbildung 12d). Sie entspricht Projektion- und Lebenszyklusvererbung.

- `PJ3`. Wie die Regel `PJ` führt auch die Transformation `PJ3` zu Projektions- und Lebenszyklusvererbung. Allerdings fügt diese Regel die zusätzlichen Aktivitäten auf einem parallelen Ausführungspfad ein (siehe Abbildung 12d).

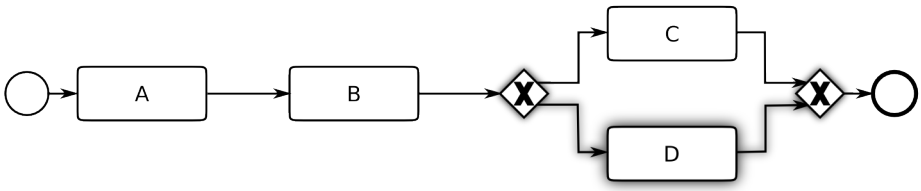
- `PP`. Diese Transformationsregel fügt neue Aktivitäten so in den Basisprozess ein, dass alle oben geschilderten Vererbungsarten zutreffen. Die beliebig oft ausführbaren Aktivitäten befinden sich auf einem zu einer Kontrollflusskante alternativen Ausführungspfad (siehe Abbildung 12e).



a) Basisprozess

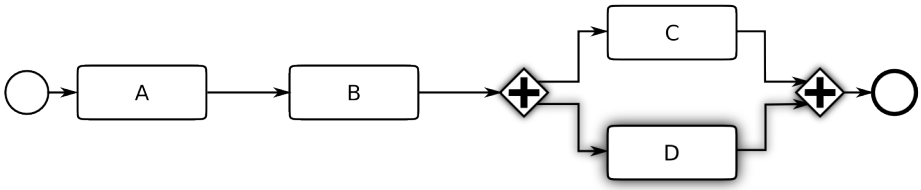


b) eigene Visualisierung der vier Transformationsregeln zum Einfügen der Aktivität D in den Basisprozess

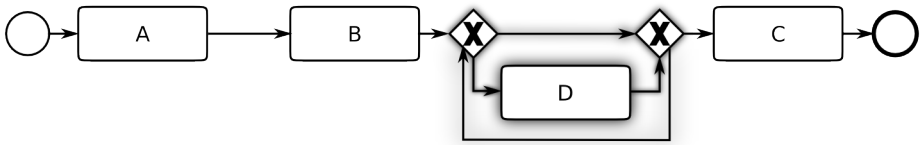


c) abgeleiteter Prozess nach Protokollvererbung sowie Lebenszyklusvererbung
unter Anwendung von Transformationsregel `PT(C)`

d) abgeleiteter Prozess nach Projektionsvererbung sowie Lebenszyklusvererbung
unter Anwendung von Transformationsregel `PJ(B,C)`



e) abgeleiteter Prozess nach Projektionsvererbung sowie Lebenszyklusvererbung
unter Anwendung von Transformationsregel `PJ3(C)`



f) abgeleiteter Prozess nach Protokoll-/Projektionsvererbung, Protkollvererbung,
Projektionsvererbung sowie Lebenszyklusvererbung
unter Anwendung von Transformationsregel `PP(B,C)`

Abbildung 12: a) Basisprozess, b) vier Transformationsregeln zum Einfügen der Aktivität D in den Basisprozess, c)-f) Prozessmodelle, bzw. Varianten nach Anwenden einer Transformationsregel

	Transformationsregeln			
	PT	PJ	PJ3	PP
Vererbungsart				
Protokollvererbung	√			√
Projektionsvererbung		√	√	√
Protokoll-/Projektionsvererbung				√
Lebenszyklusvererbung	√	√	√	√

Tabelle 1: Vererbungsarten und ihre zugehörigen Transformationsregeln

Modulare Vorgehensweise Bei der modularen Vorgehensweise wird von einem Basisprozess ausgegangen, aus dem sich durch das Anwenden von definierten Änderungen Varianten generieren lassen. Ein Beispiel für diesen Ansatz ist die in [Hal09, HBR10a, HBR10b] entwickelte Provop-Methode zum Erstellen von Prozessvarianten mittels *Optionen*. Eine Option ist eine Gruppe von Änderungsoperationen, die stets als Ganzes auf einen Basisprozess angewandt werden kann. Grundlegende *Änderungsoperationen* sind das Einfügen, Löschen, Verschieben und Modifizieren von Aktivitäten (siehe [WRRM08] für entsprechende Änderungsoptionen und [RMRW08] für deren formale Semantik). Im Basisprozess selbst werden sogenannte *Aufsetzpunkte* vorgesehen, an denen die Änderungsoperationen beginnen bzw. enden. Abbildung 13 verdeutlicht dies an einem Beispiel. Ausgangspunkt der *Variantenkonfiguration* bilden ein Basisprozess, mit den durch Rauten markierten Aufsetzpunkten, und die Optionen 1 bis 6, die der Einfachheit hier jeweils nur aus einer Änderungsoperation bestehen. Die Einfügeoperation besteht aus zwei Teilen: einem einzufügenden Prozessfragment, das durch eine Beginn- und eine Endemarkierung begrenzt wird (dargestellt durch Rauten mit der Beschriftung S respektive E) und einer Abbildung dieser Markierungen auf die gewünschten Aufsetzpunkte im Basisprozess. Das Prozessfragment wird immer auf einem parallelen Kontrollpfad in den Prozess eingefügt (vgl. Abbildung 13c). Wie in Abbildung 13d) zu sehen, resultiert dies in einem sequenziellen Einfügen, wenn der Graph entsprechend vereinfacht werden kann, ohne sein Ausführungsverhalten zu ändern. Alternative Kontrollpfade können eingefügt werden, indem die entsprechenden Verzweigungsknoten in dem einzufügenden Prozessfragment berücksichtigt werden (Abbildung 13e). Zur Definition einer Löschoperation kann die zu löschende Aktivität, oder die Aufsetzpunkte zwischen denen gelöscht werden soll, angegeben werden. Um ein Prozessfragment zu verschieben, wird der Aufsetzpunkt, an dem das zu verschiebende Prozessfragment beginnt, dem Aufsetzpunkt zugewiesen, an dem es eingefügt werden soll. Analog wird für den Endaufsetzpunkt vorgegangen. Eine weitere Änderungsoperation ist das Modifizieren von Eigenschaften einer Aktivität. Wie in Abbildung 13h) zu sehen, hat dies keinen Einfluss auf die Struktur des Prozessmodells.
Provop unterstützt zudem eine kontextabhängige Konfiguration [HBR08a, HBR09]. Dazu wird vom Modellierer ein Kontextmodell erstellt und den Optionen werden Kontextbedingungen zugewiesen. Der Modellierer erhält zudem die Möglichkeit, Kombinationen von ungültigen Kontextwerten auszuschließen und Bedingungen (Reihenfolge, Hierarchie, Implikation, Auswahl n-aus-m, wechselseitiger Ausschluss) für die Optionen zu definieren. Ist die Kontextbedingung einer Option erfüllt und liegt eine gültige Kombination von Kontextwerten vor, wird diese Option auf den Basisprozess angewandt. Weitere Details zum Provop-Ansatz finden sich in [HBR10a, HBR10b, HBR08a, HBR09].

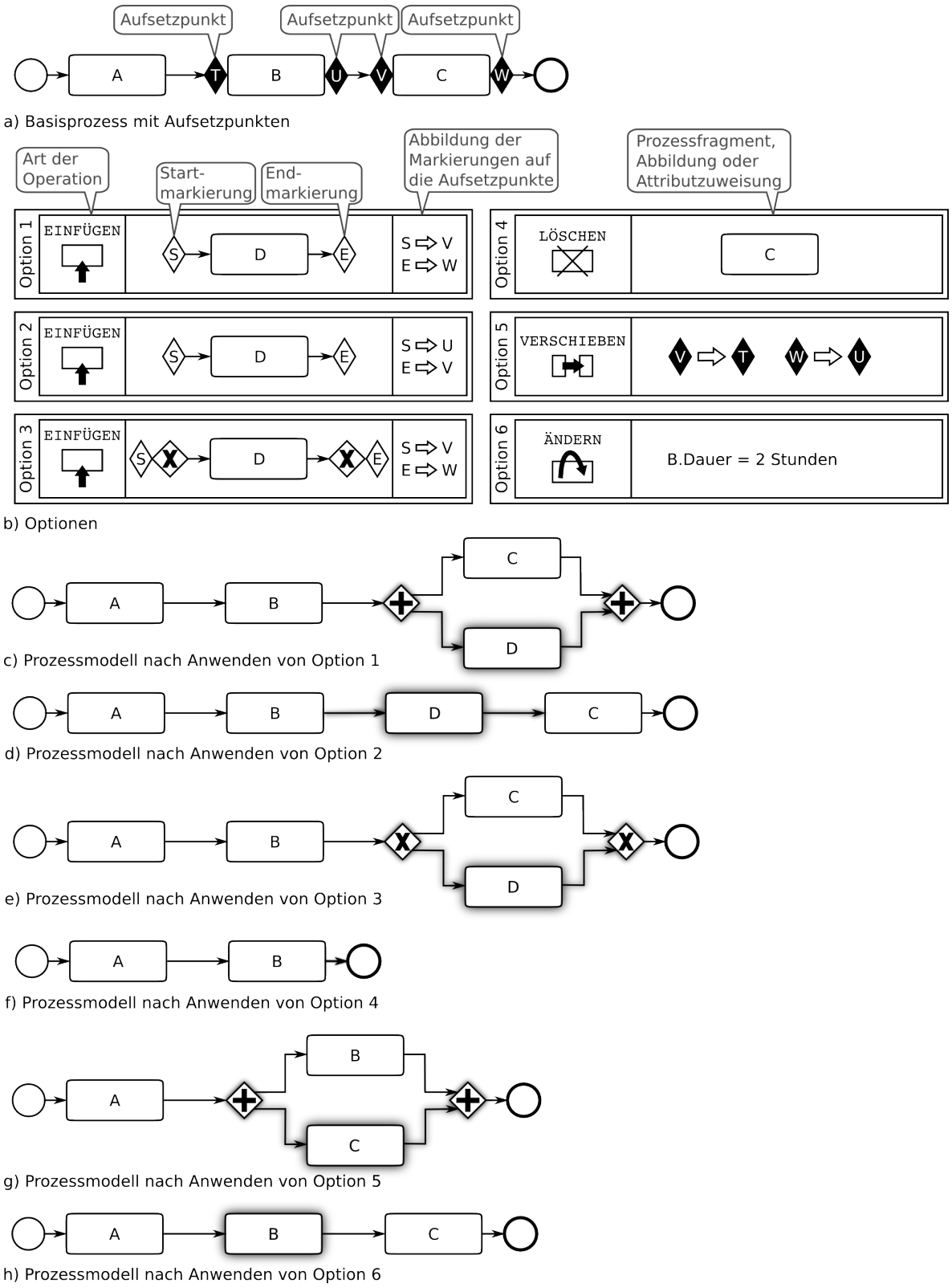


Abbildung 13: a) Basisprozess mit Aufsetzpunkten, b) Optionen mit jeweils einer Änderungsoperation, c)-h) Prozessmodelle bzw. -varianten nach Anwenden jeweils einer Option

3.2 Variantenmodellierung durch Selektion

Bei der Variantenmodellierung durch Selektion sind alle Varianten bereits im Basisprozess enthalten, d.h. der Basisprozess ist eine Obermenge aller Prozessvarianten.

Vorgehensweise mit konfigurierbarem Prozessmodell Dieser Ansatz greift die vorgestellten Vererbungsgrundsätze von [AJ00, AB02, Aal03] auf und schlägt das inverse Vorgehen vor. Der Basisprozess, der als *konfigurierbares Prozessmodell* [ADG+06, GAJVL08] bezeichnet wird, ist ein aus allen Varianten unter Beibehaltung ihres Verhaltens abgeleiteter Prozess und damit ihr kleinstes gemeinsames Vielfaches. Dies bedeutet, dass das konfigurierbare Prozessmodell in der Regel umfangreicher ist als eine konkrete Variante, d.h. es besitzt zusätzliche Kontrollflusskanten, Verzweigungsknoten und/oder Aktivitäten.
Um das konfigurierbare Prozessmodell zu erstellen, wird in [LDUD09, LDKD10] ein Mechanismus für das Zusammenführen von Prozessvarianten vorgestellt. Dieser genügt den Anforderungen, dass das entstehende Prozessmodell erstens das Verhalten aller Prozessvarianten aufweist und zweitens feststellbar ist aus welcher Variante ein Prozesselement stammt. Schließlich muss das Prozessmodell so konfigurierbar sein, dass alle ursprünglichen Varianten generiert werden können. Damit erkennbar ist aus welcher Prozessvariante ein Knoten des konfigurierbaren Prozessmodells stammt, können zwei Knoten nur zusammengefügt werden, wenn ihre Lage in den Prozessvariantenmodellen gleich ist, d.h. wenn beide Knoten den Anfang ihres Variantenprozessmodells bilden (nur Nachfolgerknoten), beide Konten das Ende des ihres Prozessmodells beschreiben (nur Vorgängerknoten) oder sich beide Knoten innerhalb ihres Prozessmodell befinden (Vorgänger- und Nachfolgerknoten).
Zur Illustration zeigt Abbildung 14 ein konfigurierbares Prozessmodell für die Prozessmodelle bzw. -varianten aus Abbildung 13c)-h). Jede Kante des Modells ist mit einer Beschriftung versehen, die darüber Aufschluss gibt, in welcher Prozessvariante sie enthalten ist. Zudem enthält das Prozessmodell sogenannten *Variationspunkte (Variation Points)* [ADG+08], an denen eine *Konfigurationsentscheidung* getroffen werden muss. Die Variationspunkte sind wie klassische Verzweigungsknoten in das Prozessmodell eingebunden, um sie dennoch von diesen unterscheiden zu können, weisen sie eine dickere Umrandung auf. Neben den in der Abbildung gezeigten konfigurierbaren Verzweigungsknoten, sind auch konfigurierbare Aktivitäten möglich.

Die Konfiguration einer Prozessvariante [GAJVL08], d.h. die Selektion aus dem Basisprozess erfolgt durch die bereits bei der objektorientierten Modellierung durch Transformation beschriebenen Ansätze des Blockierens und Versteckens (vgl. Abschnitt Objektorientierte Vorgehensweise). Dazu wird jede in einen Variationspunkt eingehende Kontrollflusskante *(Inflow)* entweder mit dem Wert zulässig *(Allowed)*, versteckt *(Hidden)* oder blockiert *(Blocked)* markiert. Ausgehende Kontrollflusskanten *(Outflow)* können die Werte zulässig oder blockiert tragen. Ist eine eingehende oder ausgehende Kontrollflusskante zulässig, wird sie in die Variante übernommen. Weist ein Variationspunkt eine versteckte eingehende Kontrollflusskante auf, so wird der Variationspunkt im entstehenden Prozessmodell ausgelassen und verhält sich selbst wie eine Kontrollflusskante, d.h. die eingehende Kante wird mit den zulässigen ausgehenden Kontrollflusskanten verbunden. Blockierte eingehende oder ausgehende Kontrollflusskanten werden nicht in die konfigurierte Variante aufgenommen. Das impliziert, dass Prozessbestandteile, die nur über eine blockierte Kontrollflusskante zu erreichen sind, nicht in die entstehende Prozessvariante aufgenommen werden.
Diesem Ansatz entsprechend wird in [LRDHM11] konkretisierend festgehalten, dass ein konfigurierbarer `XOR`-Verzweigungsknoten mit zwei ausgehenden Kontrollflusskanten so konfiguriert werden kann, dass nur eine der ausgehenden Kanten in die Variante übernommen wird, wobei der Verzweigungsknoten gleichzeitig entfernt wird. Das bedeutet, dass eine ausgehende Kontrollflusskante

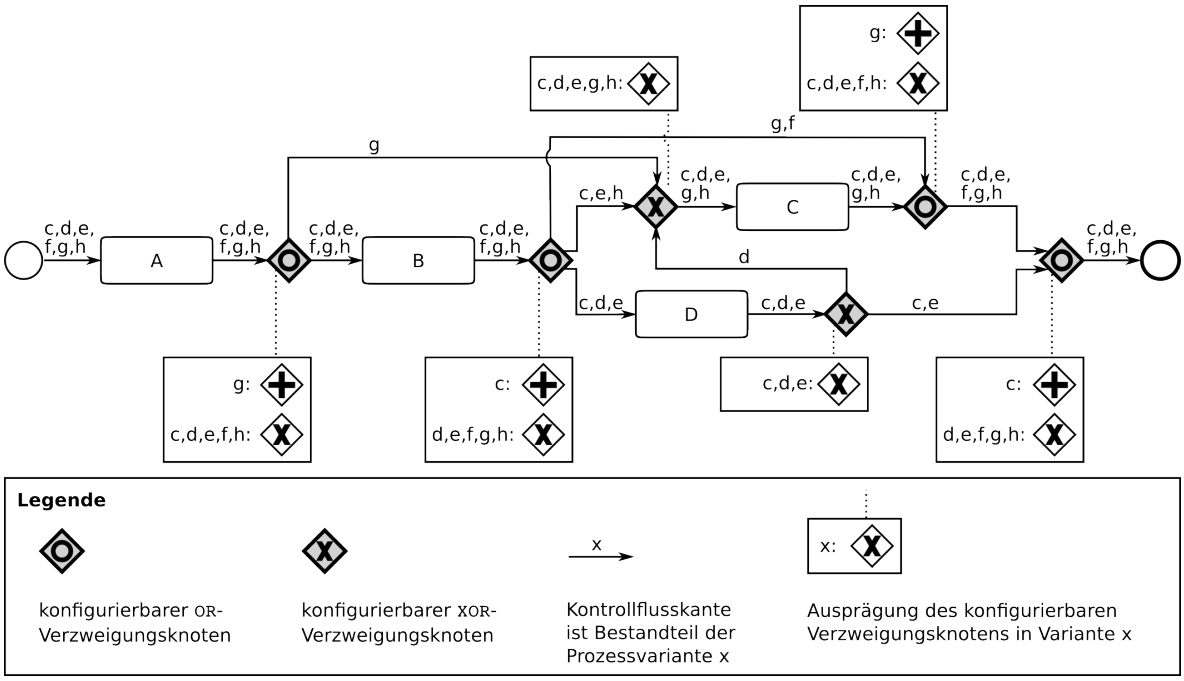


Abbildung 14: Konfigurierbares Prozessmodell für die Prozessmodelle aus Abbildung 13c)-h)

zulässig markiert wird und die andere blockiert wird. Die eingehende Kante kann dabei entweder versteckt oder zulässig markiert werden. Im zweiten Fall wird der Verzeigungsknoten zuerst mit einer eingehenden und einer ausgehenden Kontrollflusskante in die Prozessvariante aufgenommen und anschließend bei der Vereinfachung des Graphen durch eine Kontrollflusskante ersetzt. Ein konfigurierbar XOR-Verzweigungsknoten kann aber auch als regulärer XOR-Verzweigungsknoten in das Prozessmodell übernommen werden (alle Kontrollflusskanten sind zulässig), d.h. die Auswahl des Ausführungspfads wird erst zur Laufzeit getroffen. Konfigurierbare OR-Verzweigungsknoten bieten die größte Flexibilität. Besitzt der Knoten zwei ausgehende Kanten, so kann er und eine der beiden Kontrollflusskanten, analog zum XOR-Verzweigungsknoten, bei der Konfiguration aus dem Prozessmodell entfernt werden. Alternativ können alle Kontrollflusskanten in die Variante übernommen werden (alle Kanten zulässig markiert) und der konfigurierbare Knoten in einen regulären OR-, XOR- oder AND-Verzweigungsknoten umgewandelt werden.
Abbildung 15 zeigt wie aus dem konfigurierbaren Prozessmodell in Abbildung 14 die Prozessvariante aus Abbildung 13f) konfiguriert werden kann. Dabei wird beispielsweise der konfigurierbare OR-Verzweigungsknoten zwischen den Aktivitäten A und B zu einem versteckten XOR-Verzweigungsknoten mit einer ausgehenden Kontrollflusskante konfiguriert, d.h. der Knoten in der Prozessvariante durch eine Kontrollflusskante ersetzt.

Da der Konfiguration ein umfangreiches und komplexes Prozessmodell zu Grunde liegt, muss die Person, die es konfiguriert, geeignet unterstützt werden. [LADH09] schlägt dazu einen *interaktiven Fragebogen* vor, d.h. für jeden Variationspunkt werden Fragen zu den *Rahmenbedingungen (Domain Facts)* gestellt, die jeweils mit richtig und falsch beantwortet werden können. Durch Auswertung der entsprechenden Antworten wird festgestellt, welche Werte den ein- bzw. ausgehenden Kontrollflusskanten zugewiesen werden können. Die Rahmenbedingungen selbst können voneinander unabhängig oder teilweise bzw. vollständig abhängig sein. Dies wird dadurch berücksichtigt, dass zwischen den zugehörigen Fragen die entsprechenden Beziehungen modelliert werden. Beziehen sich zwei Fragen

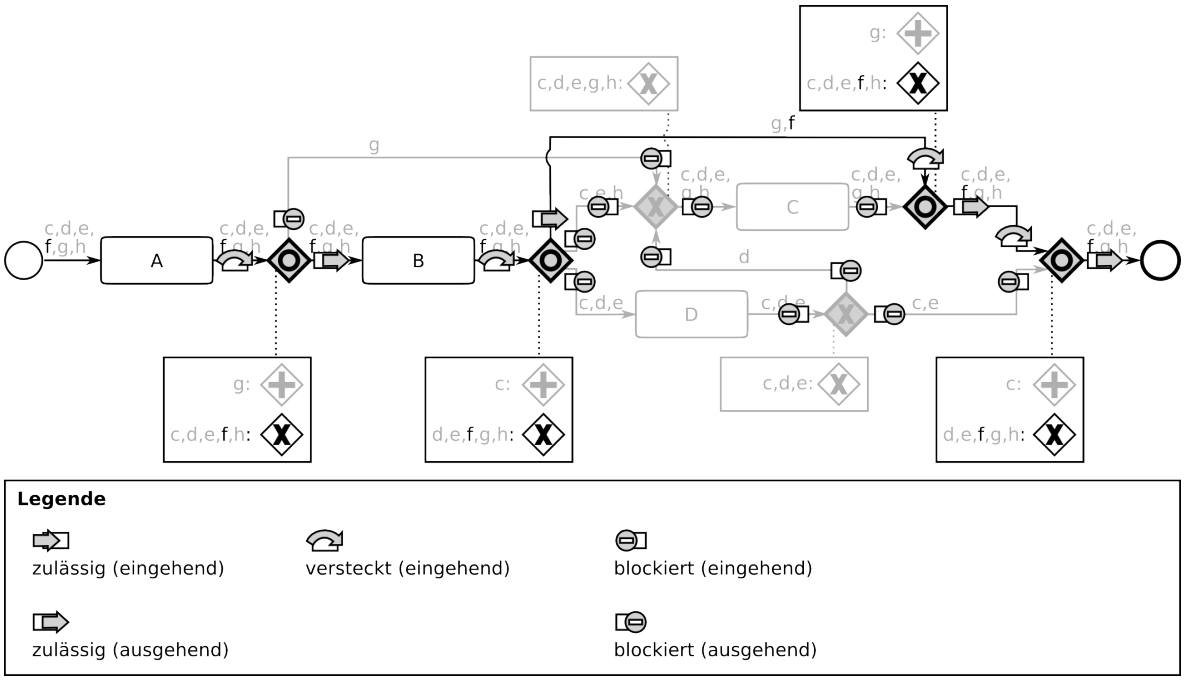


Abbildung 15: Konfiguration des Prozessmodells aus Abbildung 13f)

auf dieselbe Rahmenbedingung, ist die erste Antwort ausschlaggebend, d.h. sie ist gleichzeitig für die zweite Frage gültig.
Um die syntaktische und semantische Korrektheit des konfigurierten Prozessmodells sicherzustellen, wird in [ADG+10] ein Regelwerk definiert, das eine Korrektheitsprüfung in jedem Konfigurationsschritt, d.h. an jedem Variationspunkt, erlaubt.

Weiterführende Betrachtungen zum konfigurierbaren Prozessmodell ergänzen dieses um Daten und Rollen [LDH+08, LRDHM11]. Wie in Abbildung 16 zu sehen, können für Datenobjekte und Rollen diverse Beziehungen zu einer Aktivität definiert werden, indem sie entweder direkt oder über einen Verzweigungsknoten, der eine logische Verknüpfung zwischen Datenobjekten bzw. Rollen festlegt, mit der Aktivität verbunden werden. Zu den logischen Verknüpfungen zählt neben `und`, `oder` sowie `exklusiv oder` auch die Möglichkeit, die Kardinalität der Auswahl festzulegen. Darüber hinaus kann die Verbindungskante zu einer Aktivität als optional gekennzeichnet werden. Bei den Datenobjekten wird zudem zwischen in die Aktivität eingehenden und aus der Aktivität ausgehenden Datenflusskanten unterschieden. Erstere bedeuten, dass das Datenobjekt der Aktivität zur Verfügung gestellt wird, letztere gibt wieder, dass das Datenobjekt von der Aktivität erzeugt wird. Der Datenfluss zwischen den Aktivitäten wird implizit modelliert, d.h. tragen zwei Datenobjekte denselben Namen, so handelt es sich um dasselbe Datenobjekt. Zudem wird bei diesem Modellierungsansatz berücksichtigt, dass ein Eingabedatenobjekt von einer Aktivität verbraucht werden kann, d.h. es kann anderen Aktivitäten nicht mehr zur Verfügung gestellt werden. Zusätzlich zum Prozessmodell werden sowohl zwischen den Datenobjekten als auch zwischen den Rollen Hierarchiebeziehungen bzw. Spezialisierungsbeziehungen definiert. In Abbildung 16b) wird beispielhaft die Hierarchiebeziehung der Rollen illustriert.

In einem erweiterten konfigurierbaren Prozessmodell können neben Aktivitäten und Verzweigungsknoten auch Datenobjekte und Rollen als konfigurierbar markiert werden. Dabei lassen sich

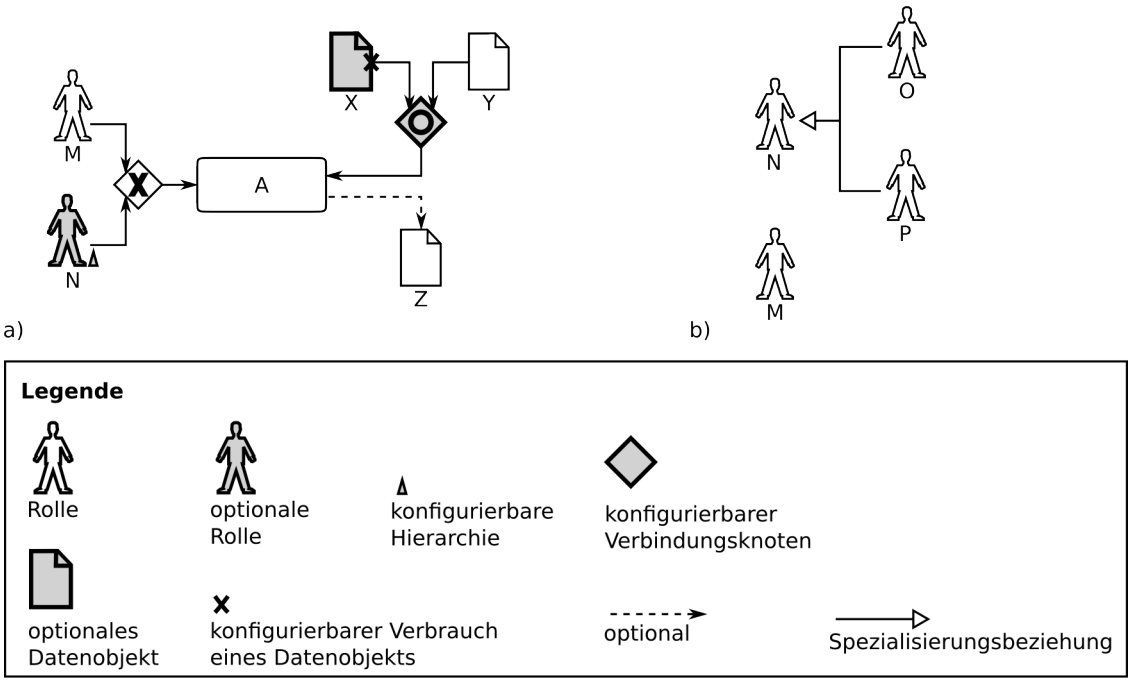


Abbildung 16: a) Erweiterung eines konfigurierbaren Prozessmodells um Datenobjekte und Rollen am Beispiel einer einzelnen Aktivität, b) Hierarchie- bzw. Spezialisierungsbeziehung der Rollen

zwei voneinander unabhängige Konfigurationsdimensionen unterscheiden: Optionalität und Hierarchiebeziehungen/Spezialisierungen. Für ein optionales Datenobjekt bzw. eine optionale Rolle kann in der Konfigurationsphase entschieden werden, ob es in das Variantenprozessmodell übernommen wird. Ist eine Hierarchiebeziehung für das Datenobjekt bzw. die Rolle hinterlegt, so kann eine Spezialisierung vorgenommen werden. Konfigurierbare Eingabedatenobjekte können zudem von verbraucht *(Consumed)* zu benutzt *(Used)* konfiguriert werden. Des Weiteren kann die Kardinalität der Auswahl bei der Konfiguration eingeschränkt werden, d.h. die neue Auswahlbedingung liegt innerhalb des ursprünglichen Intervalls.
Die Konfiguration findet wieder mittels Fragebogen statt. Dabei werden Abhängigkeiten zwischen Aktivitäten, konfigurierbaren Verzweigungsknoten, Datenobjekten und Rollen berücksichtigt, um ein korrekt konfiguriertes Prozessmodell zu erhalten.

Vorgehensweise mit Fragmenten und Bedingungen Ein weiterer Ansatz zu Variantenmodellierung, das *Business Process Constraint Network (BPCN)*, wird in [SOS05, LSG09] vorgestellt. Der Basisprozess dieses Ansatzes zeichnet sich durch einen für alle Varianten gleichen *Prozesskern* sowie *flexiblen Prozessfragmenten* aus. Letztere sind eine Menge von Aktivitäten, auf denen *Modellierungsbedingungen* definiert werden. Bei der Variantenmodellierung werden unter den erwähnten Bedingungen eine Auswahl aus den Aktivitäten getroffen und ihre Anordnung festgelegt. Als Grundlage der Variantenmodellierung müssen jedoch nicht zwingend alle drei Bestandteile (Kernprozess, Prozessfragmente und Modellierungsbedingungen) vorgegeben werden. Es genügt auch die alleinige Modellierung des Basisprozesses, d.h. die Vorgabe von Prozesskern und -fragmenten. Durch das Fehlen von Bedingungen auf den Fragmenten wird eine große Flexibilität in der Variantenmodellierung erreicht. Ebenso ist es möglich, nur Prozessfragmente und Modellierungsbedingungen zu definieren,

d.h. die entstehenden Varianten werden nicht zwingend einen gemeinsamen Prozesskern aufweisen. Schließlich wird die maximale, aber am wenigsten steuerbare Flexibilität zur Variantenmodellierung angeboten, wenn nur Prozessfragmente zur Verfügung gestellt werden, d.h. diese frei wählbar und kombinierbar sind.

Die Modellierung des Prozesskerns unterscheidet sich nicht von der Modellierung eines nicht variablen Prozesses. Jedoch werden in den Prozesskern an der Stelle, die variabel gestaltet werden soll, Prozessfragmente eingefügt. Prozessfragmente sind Aktivitäten oder Subprozesse, die als Menge in den Kernprozess eingebettet werden, d.h. zwischen ihnen ist kein Kontrollfluss vorgegeben (siehe Abbildung 17a). Der Kontrollfluss zwischen den Fragmenten wird zur Konfigurations- oder Laufzeit anhand der Modellierungsbedingungen festgelegt. Diese Bedingungen lassen sich in Auswahl- und Ablaufbedingungen unterscheiden. Die Auswahlbedingungen legen fest, welche Fragmente in die Prozessvarianten aufgenommen werden (siehe Tabelle 2). Die acht Ablaufbedingungen bezeichnen hingegen Kontrollflusskonstrukte und zeitliche Abhängigkeiten zwischen den Aktivitäten (siehe Tabelle 3).

Auswahlbedingung	Beschreibung
obligatorische Auswahl *(Mandatory Constraint)*	Legt Fragmente fest, die zwingend ausgeführt werden müssen.
verbotene Auswahl *(Prohibitive Constraint)*	Bezeichnet Aktivitäten, die nicht ausgeführt werden dürfen.
Auswahlumfang *(Cardinality Constraint)*	Legt fest, wie viele Aktivitäten mindestens und wie viele höchstens aus der Menge der Aktivitäten ausgewählt werden können.
Inklusionsauswahl *(Inclusion Constraint)*	Beschreibt, wenn eine Aktivität eine andere Aktivität zwingend nach sich zieht.
Voraussetzungsauswahl *(Prerequisite Constraint)*	Ist das Inverse der Inklusionsauswahl, d.h. wird eine Aktivität nicht ausgewählt, so darf eine andere bezeichnete Aktivität ebenfalls nicht in die Prozessvariante aufgenommen werden.
Exklusionsauswahl *(Exclusion Constraint)*	Beschreibt, dass sich zwei Aktivitäten gegenseitig ausschließen.
Substitutionsauswahl *(Substitution Constraint)*	Legt fest, dass eine Aktivität eine andere nicht gewählte Aktivität substituiert.
zugleiche Auswahl *(Corequisite Constraint)*	Definiert, dass entweder beide oder keine der angegebenen Aktivitäten ausgeführt werden darf.
ausschließliche Auswahl *(Exclusive-Choice Constraint)*	Legt fest, dass genau eine der angegebenen Aktivitäten in einer Variante ausgeführt werden muss.

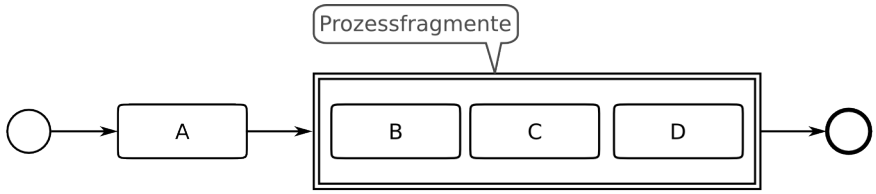
Tabelle 2: Die neun Auswahlbedingungen des Business Process Constraint Network

In Abbildung 17b) wurden die *obligatorische Auswahl* und der *Auswahlumfang* als Auswahlbedingungen definiert. Dadurch wird zum einen festgelegt, dass die Aktivität B zwingend in jedem

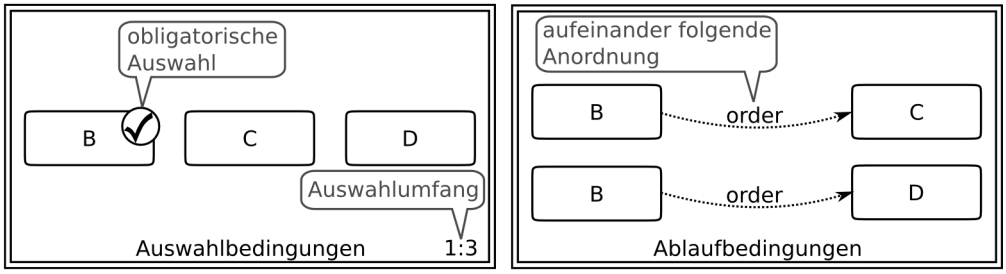
Ablaufbedingungen	Beschreibung
zuvor Anordnung *(Before Constraint)*	Legt fest, dass eine Aktivität sequentiell vor der anderen ausgeführt wird, wobei sie nicht direkt aufeinander folgen.
aneinander anschließende Anordnung *(Meet Constraint)*	Beschreibt zwei direkt aufeinander folgende Aktivitäten.
aufeinander folgende Anordnung *(Order Constraint)*	Legt allgemein fest, in welcher Reihenfolge die bezeichneten Aktivitäten ausgeführt werden.
zugleich beginnende Anordnung *(Starts Constraint)*	Bezeichnet eine Aktivität, die eine andere parallel startet, wobei sie vor der parallel gestarteten endet.
zugleich endende Anordnung *(Finishes Constraint)*	Beschreibt eine Aktivität, die gleichzeitig mit einer parallelen Aktivität endet, wobei sie später als die parallele Aktivität gestartet wurde.
währenddessen Anordnung *(During Constraint)*	Definiert eine Aktivität, die während der Ausführung einer parallelen Aktivität gestartet und beendet wird.
gleichzeitige Anordnung *(Equals Constraint)*	Beschreibt, dass zwei parallele Aktivitäten zeitgleich gestartet und beendet werden.
parallele Anordnung *(Parallel Constraint)*	Legt fest, dass zwei Aktivitäten zueinander parallel sind, ohne Aussagen über Start und Ende der Aktivitäten zu treffen.

Tabelle 3: Die acht Ablaufbedingungen des Business Process Constraint Network

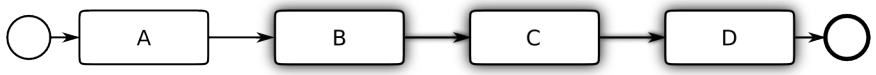
konfigurierten Prozessmodell berücksichtigt werden muss. Zum anderen wird vorgegeben, dass mindestens eine, aber höchstens drei der vorgegebenen Aktivitäten in einer Prozessvariante berücksichtigt werden müssen. Die definierten Ablaufbedingungen geben vor, dass die Aktivität B vor den Aktivitäten C und D ausgeführt werden muss. Mit diesen Bedingungen lassen sich neben den Prozessmodellen in Abbildung 17c)-d) auch die Prozessmodelle aus Abbildung 13c)-f) und h) generieren. Das Prozessmodell aus Abbildung 13g) in lässt sich nicht erstellen, da die Ablaufbedingung, dass Aktivität B vor Aktivität C ausgeführt werden muss, verletzt wird.



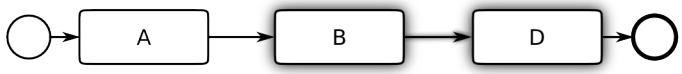
a) Kernprozess und Prozessfragmente



b) Modellierungsbedingungen



c) mögliches Prozessmodell



d) mögliches Prozessmodell

Abbildung 17: a) Kernprozess mit Prozessfragmenten, b) Auswahl- und Ablaufbedingungen, c)-d) mögliche Prozessmodelle

3.3 Modellierung eines Basisprozesses

Der Basisprozess ist der Ausgangspunkt aller vorgestellten Ansätze und spielt aus diesem Grund eine wichtige Rolle. Es existieren mehrere Möglichkeiten, ihn zu modellieren. Er kann den *Standardprozess*, die *am häufigsten ausgeführte Prozessvariante*, die *Schnittmenge aller Prozessmodelle*, die *Obermenge aller Prozessmodelle* oder die *minimale durchschnittliche Modelldifferenz* darstellen [vgl. Hal09, HBR10b]. Im objektorientierten Ansatz (vgl. Abschnitt Objektorientierte Vorgehensweise) wird typischerweise die Schnittmenge aller Prozessvarianten gewählt, da die Transformationsregeln ausschließlich Prozesselemente hinzufügen, nicht aber löschen können. Bei der Provop-Methode

(vgl. Abschnitt Modulare Vorgehensweise) kann die Art des Basisprozesses frei gewählt werden. Werden Prozessvarianten durch Selektion aus dem Basisprozess erstellt, so ist letzterer eine Obermenge aller Prozessmodelle.
Im Folgenden soll exemplarisch dargestellt werden, wie ein Basisprozess modelliert werden kann. Nachdem bereits beschrieben wurde, wie anhand verschiedener Ansätze eine Obermenge der Varianten erzeugt werden kann, soll ein Ansatz zur Modellierung der minimal durchschnittlichen Modelldifferenz betrachtet werden.

In MinAdept [LRW08a, LRW09a, LRW09b, LRW10] werden Algorithmen vorgestellt, der auf den (blockstrukturierten) Prozessvarianten und ihrer Ausführungshäufigkeit basierend einen Basisprozess erzeugt. Von dem so erstellten (blockstrukturierten) Basisprozess lassen sich die Prozessvarianten im Durchschnitt mit möglichst wenigen Änderungsoperationen, wie Einfügen, Löschen und Verschieben einer Aktivität, realisieren. Um diesen Basisprozess zu erzeugen, wird jede Prozessvariante in einer zweidimensionalen Matrix, der sogenannten *Order Matrix*, abgebildet. Die beiden Dimensionen repräsentieren jeweils alle im Prozess vorhandenen Aktivitäten, die Zellen beschreiben die (transitiven) Beziehung der Aktivitäten zueinander. Entweder sind die beiden betrachteten Aktivitäten sequenziell angeordnet, so dass sie zueinander in einer Vorgänger- oder Nachfolgerbeziehung stehen, oder sie bilden einen `AND`- oder `XOR`-Block, d.h. sie befinden sich auf parallelen oder alternativen Ausführungspfaden zueinander. Die so erhaltenen Variantenmatrizen, d.h. die Order Matrizen aller Prozessvarianten, werden anschließend zu einer Matrix, der sogenannten *Aggregierten Order Matrix*, aggregiert. Das bedeutet, dass die Informationen in einer Zelle dieser Matrix nicht mehr eindimensional sind, sondern einen Vektor mit den Dimensionen Vorgänger, Nachfolger, `AND`-Block und `XOR`-Block beschreiben. Die Werte des Vektors geben dabei an, wie häufig diese Beziehung über alle Varianten summiert auftritt. In einem weiteren Schritt werden alle Aktivitäten iterativ in Blöcke zusammengefasst. Dazu werden Paare von Aktivitäten verglichen. Weisen zwei Aktivitäten dieselbe Anordnungsbeziehung zu den restlichen Aktivitäten auf, so bilden sie einen Block. Um dies zu beurteilen wird eine Separationstabelle *(Separation Table)* erzeugt, die für jedes Aktivitätenpaar den sogenannten Separationswert *(Separation Value)* enthält. Dieser berechnet sich, indem alle Vektoren erzeugt werden, die von den Aktivitäten des untersuchten Paares ausgehen und in einer der verbliebenen Aktivitäten enden. Die Vektorenpaare, die in derselben Aktivität enden, werden auf ihren Abstand zueinander untersucht. Alle Abstände werden für ein Aktivitätenpaar aufsummiert und normalisiert. Das Paar mit dem geringsten Abstand wird als Block modelliert. Die Anordnung der Aktivitäten innerhalb des Blocks wird festgelegt, indem untersucht wird, zu welcher Koordinatenachse der Vektor des Paares den geringsten Abstand aufweist. Schließlich wird die aggregierte Matrix unter Berücksichtigung des gebildeten Blocks neu berechnet und der nächste Iterationsschritt eingeleitet.

3.4 Erkennen von Varianten

Verschiedene Ansätze [WW10, APW08, DDGB09, DDM08, LRW08b] beschäftigen sich damit, Prozessvarianten zu identifizieren, wobei dies oft als *Bestimmung der strukturellen und/oder semantischen Ähnlichkeit von Prozessen* bezeichnet wird. Durch die Analyse der Ähnlichkeit können einerseits redundante Prozesse vermieden werden, andererseits können Varianten erkannt und explizit als solche modelliert werden. Letzteres bringt Vorteile, etwa die einmalige Modellierung gleicher Prozessabschnitte, gleiche Bezeichnungen für semantisch identische Aktivitäten oder auch die gemeinsame Evolution aller Varianten eines Basisprozesses mit sich.
Da das Erkennen und damit das Speichern und Suchen von Varianten nicht im Fokus dieser Arbeit liegt, soll dies im Folgenden nur kurz am Beispiel des *Process Variants Repository (PVR)* [LS06,

LSG09] vorgestellt werden. Dieses baut zwar auf dem Prinzip der Variantenmodellierung nach dem Business Process Constraint Network (vgl. Abschnitt Vorgehensweise mit Fragmenten und Bedingungen) auf, die Speicherung und das Suchen der Prozessvarianten aufgrund von Ähnlichkeitskriterien sowie das Bewerten der Ergebnisse in Form einer Rangfolge ist, jedoch auch auf Prozesse im Allgemeinen anwendbar.
Im Process Variants Repository werden alle Prozesse nach ihrer Ausführung gespeichert. Dazu werden jeweils eine erläuternde und für den Modellierer verständliche Beschreibung, Informationen zur Prozessausführung (Ausführungsreihenfolge, Laufzeit, Ressourcen) und die Struktur der Prozessvariante abgelegt. Diese Informationen werden als Schema der Prozessvariante *(Process Variant Schema)* bezeichnet.
Ausgangspunkt einer Suche nach eine Prozessvariante sind eine Suchanfrage, die aus mehreren Teilen bestehen kann, und der Speicher in dem jede Variante durch ein Prozessvariantenschema beschrieben ist. Eine mehrteilige Suchanfrage wird iterativ durchgeführt, d.h. zuerst wird das Ergebnis des ersten Anfragebestandteils aus der Menge aller Schemata, d.h. dem gesamten Speicher, generiert. Anschließend wird jedes weitere Anfragebestandteil in der Ergebnismenge gesucht, die die jeweils direkt vorangegangene Suche generiert hat.
Die Suche auf dem Variantenspeicher wird mit zwei Techniken realisiert. Einerseits lassen sich viele Suchen mit der *Structured Query Language (SQL)* beschreiben, andererseits genügen deren Mechanismen nicht, um eine Suche nach einer bestimmten Prozessstruktur vorzunehmen. Zu diesem Zweck wird nach dem Prinzip *Query by Example (QBE)* vorgegangen, d.h. die gesuchte Struktur graphisch modelliert. Dabei kann diese Struktur einen Ausschnitt aus einem Prozessmodell oder ein vollständiges Prozessmodell darstellen. Die Suche nach einer Variante soll nicht nur exakte Ergebnisse liefern, sondern auch ähnliche Prozessvarianten liefern, wobei letztere nach ihrer Ähnlichkeit zum gesuchten Prozessmodell geordnet sein sollen. Die Ähnlichkeit zwischen einem gesuchten Prozessmodell und den im Speicher abgelegten Varianten wird unter den Gesichtspunkten der Struktur und der Semantik betrachtet. Das bedeutet, dass sowohl der Aufbau des Prozessmodells, als auch das Verhalten bei der Ausführung, d.h. die Ausführungsreihenfolge der Aktivitäten, verglichen werden.
Es werden drei Kategorien struktureller Ähnlichkeit unterschieden: strukturell äquivalent, strukturell subsumiert und strukturell impliziert. Strukturelle Äquivalenz, d.h. eine vollständige Übereinstimmung, liegt vor, wenn alle Knoten und Kanten einer gegebenen Struktur mit alle Knoten und Kanten einer Prozessvariante identisch sind. Ist die Menge der Knoten einer Struktur eine Teilmenge der Knoten einer Prozessvariante und entsprechen die strukturellen Bedingungen zwischen den Konten der Variante denen der gegebenen Struktur, so subsumiert die Variante diese Struktur. Eine Prozessvariante impliziert eine Struktur, wenn die strukturellen Bedingungen zwischen den Konten einer Variante mit denen der gegebenen Struktur übereinstimmen und die Knotenmengen identisch sind. Eine vollständige Übereinstimmung zwischen gesuchter Struktur und einer Prozessvariante liegt vor, wenn diese strukturell äquivalent sind oder die Prozessvariante die Struktur subsumiert. Impliziert eine Variante die gesuchte Struktur, so liegt eine nahezu vollständige Übereinstimmung vor.
Zur Feststellung der Ähnlichkeit zwischen einem gesuchten Prozessmodell und den abgelegten Prozessvarianten wird im ersten Schritt die *strukturelle Ähnlichkeit* bestimmt. Dazu werden die Graphen der Prozessvarianten auf die Knoten reduziert, die auch im gesuchten Prozessmodell vorhanden sind. Als Ergebnis dieser Analyse werden zum einen die Varianten identifiziert, deren Prozessmodell vollständig mit der gesuchten Struktur übereinstimmt und zum anderen die Varianten, deren Modell nahezu übereinstimmt. Für letztere wird die *semantische Ähnlichkeit* bestimmt. Dafür wird die Aktivitätenausführungsreihenfolge der Varianten mit der potentiellen Ausführungsreihenfolge des gesuchten Prozessmodells verglichen. Dabei wird für jede Aktivität einer Ausführungsreihenfolge festgestellt, welche Aktivitäten sie in den zugehörigen Varianten auslösen kann. Die Schnittmenge zwischen dieser

Ergebnismenge und der Menge der Aktivitäten, die dieselbe Aktivität in dem gesuchten Prozessmodell auslösen kann, wird zur letzteren ins Verhältnis gesetzt und über alle Aktivitäten der Sequenz kumuliert. Anschließend wird dies mit dem Quotienten aus der Häufigkeit der Ausführungssequenz und ihrer Länge gewichtet. Der so erhaltene Wert wird schließlich mit der Anzahl der Ausführungssequenzen skaliert und beschreibt die Ähnlichkeit der Variante mit dem gesuchten Prozessmodell.

3.5 Zusammenfassung

Wir haben die Variantenmodellierung durch Transformation sowie Selektion und Ansätze zum Erkennen von Varianten anhand ihrer strukturellen und semantischen Ähnlichkeit vorgestellt.

Eine Möglichkeit, Varianten zu modellieren, stellen Transformationsregeln dar, die dem objektorientierten Paradigma der Vererbung nachempfunden sind [AJ00, AB02, Aal03]. Die Modellierung von Varianten als abgeleitete Prozesse des Basisprozesses ist jedoch eingeschränkt, da davon ausgegangen wird, dass die Aktivitäten des übergeordneten Prozesses stets auch im abgeleiteten Prozess vorhanden sind, d.h. das Löschen und Modifizieren von Aktivitäten ist nicht möglich. Des Weiteren werden die Daten und der Datenfluss in diesem Ansatz nicht explizit betrachtet, es wird lediglich angemerkt: „For the information and operation perspectives the traditional inheritance concepts are applicable.“ [AJ00, S. 274]. In [AWW03] werden weitere Aspekte formuliert, die in Zukunft bei der Modellierung von Varianten nach objektorientierten Prinzipien berücksichtigt werden können: *Overloading* und *Late Binding*. Das bedeutet, dass verschiedene Varianten modelliert werden, aber erst zur Laufzeit anhand von dann vorliegenden Informationen entschieden wird, welche Variante ausgeführt wird.
Der Provop-Ansatz, ein Beispiel der modularen Vorgehensweise, stellt Möglichkeiten zur Verfügung, um aus einem Basisprozess mittels gruppierten Änderungsoperationen, die als Optionen bezeichnet werden, Varianten zu generieren [Hal09, HBR10a, HBR10b, HBR09]. Der gesamte Lebenszyklus von Basisprozess und Optionen wird unterstützt, insbesondere wird eine automatische kontextabhängige Konfiguration ermöglicht. Der Ansatz fokussiert dabei stark auf den Kontrollfluss und berücksichtigt Daten und Rollen eines Prozesses nicht in ausreichendem Maß.

Die Konfiguration des Basisprozesses in Form von Selektion wird beim Ansatz des konfigurierbaren Prozessmodells *(Configurable Process Model)* verfolgt. Der Basisprozess ist hier eine Obermenge aller Prozessvarianten und stellt somit ein umfassendes und komplexes Prozessmodell dar, dass auch redundante Prozessfragmente aufweisen kann. Aus diesem Modell wird in der Konfigurationsphase mithilfe eines Fragebogens eine Prozessvariante erzeugt. Aktuelle Betrachtungen erweitern diesen kontrollflussfokussierten Ansatz um Datenobjekte und Rollen.
Ebenfalls ein auf Selektion basierender Ansatz ist das Business Process Constraint Network. Durch die Selektion und Anordnung von Prozessfragmenten im Rahmen der vorgegebenen Modellierungsbedingungen wird eine sehr flexible Variantenmodellierung erreicht. Die Betrachtungen fokussieren dabei auf den Kontrollfluss.

Die Ähnlichkeit von Prozessen und damit das Speichern und Suchen von Prozessen wird ebenfalls auf Ebene der Aktivitäten und des Kontrollflusses definiert.

Wie die Übersicht zeigt, steht der Kontrollfluss im Fokus der bisherigen Ansätze zur Variantenmodellierung und -erkennung. Nur der Ansatz des konfigurierbaren Prozessmodells bezieht bisher Daten und Rollen in die Betrachtung mit ein.

4 Datenflussvarianten

Operative Datenobjekte (vgl. Abschnitt Datenkategorisierung) fließen durch einen Prozess, indem sie Aktivitäten und Verzweigungsknoten zur Verfügung gestellt werden. Dieser Datenfluss wird separat vom Kontrollfluss modelliert.
Entlang den drei Segmentierungsdimensionen *Komplexität*, *Kundengruppen* und *Rahmenbedingungen* können Varianten eines Prozesses entstehen, d.h. der Kontrollfluss, der Datenfluss, die beteiligten Rollen oder Kombinationen dieser Prozessbestandteile können im Vergleich zum Ausgangsprozess eine Reihe von Anpassungen erfahren.
Von einer *Datenflussvariante* sprechen wir dann, wenn sich die betrachteten Varianten in ihrem Datenfluss unterscheiden. Darüber hinaus existieren *Kontrollflussvarianten* und *organisatorische Varianten*. Diese zeichnen sich dadurch aus, dass die Prozessvarianten in ihrem Kontrollfluss bzw. in den beteiligten Rollen voneinander abweichen. In der Regel beschränken sich die Unterschiede zwischen Prozessvarianten jedoch nicht nur auf eine Workflow Perspektive, sondern sind eine Kombination von Abweichungen in Kontrollfluss, Datenfluss und beteiligten Rollen.

Im Folgenden soll betrachtet werden, wie Datenflussvarianten entstehen können. Neben der Anpassung des Datenflusses selbst, können Modifikationen anderer Workflow Perspektiven ebenfalls zu Datenflussvarianten führen. Dies soll anhand des Beispiels „Bewerbung auf Zulassung zu einem Studiengang“ erläutert werden. Es berücksichtigt den Kontroll- und Datenfluss sowie die beteiligten Rollen.

4.1 Beispiel Bewerbung auf Zulassung zu einem Studiengang an einer Hochschule

Prozesse einer Hochschule bzw. einer Universität sind sehr vielfältiger Natur: einerseits gibt es sehr individuelle Prozesse wie die Forschung der Wissenschaftlerinnen und Wissenschaftler, andererseits gibt es Prozesse wie die Bewerbung auf Zulassung zu einem Studiengang, die für sehr viele Personen gleich sind.
Auf die Betrachtung individueller Prozesse soll an dieser Stelle verzichtet werden. Der Fokus liegt hingegen auf repetitiven Prozessen, die häufig durchgeführt werden. Diese finden sich einerseits in den *Leistungsprozessen* und andererseits in den *Unterstützungsprozessen* einer Universität.
Leistungsprozesse werden dezentral in den Fakultäten bzw. Fachbereichen, Instituten und Lehrstühlen erbracht. Die Universität gibt lediglich den Orientierungsrahmen vor, in dem Forschung, Lehre, Weiterbildung und Dienstleistungen angeboten werden. Beispielsweise erstellen die Fakultäten bzw. Fachbereiche jeweils ihre eigenen Studienordnungen und Prüfungsreglemente, und sie legen teilweise spezielle fachliche Zulassungsbedingungen fest. Diese Regelungen orientieren sich an den universitären Rahmenvorgaben, vergleicht man aber die Bestimmungen verschiedener Einheiten miteinander, können sie sehr unterschiedlich ausgestaltet sein. Dies schlägt sich aus gesamtuniversitärer Sicht in Prozessvarianten nieder, die jeweils an die konkreten Bestimmungen und Bedürfnisse einer Einheit angepasst sind. Aber auch innerhalb einer Fakultät bzw. eines Fachbereichs können mehrere Varianten für den gleichen Prozess existieren, beispielsweise wenn bei einer Fusion von zwei Fakultäten bzw. Fachbereichen (zumindest vorerst) alle bestehenden Regelungen beibehalten werden. Darüber hinaus kann eine Fakultät von vorne herein verschiedene Varianten für einen Prozess vorsehen. Beispielsweise wurde an der vierten Folgekonferenz von Bologna in London konstatiert, dass sich die beteiligten Staaten „des Werts der Entwicklung und Erhaltung einer breiten Vielfalt an Promotionswegen bewusst (seien), die auf den übergreifenden Qualifikationsrahmen für den EHR (Europäischen Hochschulraum) Bezug nehmen, wobei eine Überregulierung zu vermeiden ist“ [vgl. Eur07, S. 5].

Die angesprochene Vielfalt kann sich darin niederschlagen, dass sowohl Doktoratsprogramme mit curricularen Anteilen angeboten werden, als auch das klassische Doktorat. Das bedeutet, dass eine Fakultät zwei Promotionsvarianten unterstützen kann.
Aber auch bei den Unterstützungsprozessen, die von der zentralen Universitätsverwaltung verantwortet werden, gibt es Prozesse, die mehrere Varianten besitzen, wie beispielsweise die „Bewerbung auf Zulassung zu einem Studiengang an einer Hochschule“. Für die Durchführung des Prozesses ist zwar die zentralen Universitätsverwaltung verantwortlich, die dezentralen Organisationseinheiten beeinflussen den Prozess jedoch maßgeblich auf der Studienstufe Master und Doktorat, aber auch bei der Zulassung in ein höheres Semester auf Bachelorstufe. In diesen Fällen werden für die Zulassung oft nicht nur formelle Voraussetzungen geprüft, sondern auch bisher erworbene fachliche Qualifikationen beurteilt. Bereits diese Überlegungen zeigen, dass es nicht nur einen Prozess, sondern zumindest so viele Varianten wie Studienstufen gibt. Da es darüber hinaus jedoch weitere relevante Kriterien gibt, die den Prozess beeinflussen und die teilweise miteinander kombiniert werden können, resultieren weitere mögliche Varianten. Welche Variante jeweils zu Anwendung kommt, hängt dabei von verschiedenen Faktoren ab. Der Zulassungsprozess kann durch folgende Aspekte beeinflusst werden:

- *Zulassungsbeschränkung/Zulassungsfreiheit.* Um in stark nachgefragten Fächern die Anzahl der Studierenden zu beschränken, können für die betroffenen Studiengänge bundesweite oder lokale Zulassungsbeschränkungen erlassen werden. Die Stiftung für Hochschulzulassung, sie ist seit 1. Mai 2010 die Nachfolgeeinrichtung der Zentralstelle für die Vergabe von Studienplätzen (ZVS), ist für die Abwicklung der Bewerbungen für Studiengänge mit bundesweiter Zulassungsbeschränkung zuständig. Bei den betroffenen Studiengängen handelt es sich um Medizin, Zahnmedizin, Tiermedizin und Pharmazie [vgl. Sti10]. Lokale Zulassungsbeschränkungen zu erlassen, liegt im Ermessen der jeweiligen Universität. Dementsprechend kann jede Universität selbst entscheiden, welches Auswahlverfahren für diese Studiengänge angewandt werden soll. Es können beispielsweise dieselben Kriterien wie bei den bundesweit beschränkten Studiengängen angewandt werden, oder Bewerbungsgespräche, spezielle Vorkenntnisse und Eignungstests verlangt werden. Das Auswahlverfahren kann von der Universität selbst durchgeführt oder an die Stiftung für Hochschulzulassung delegiert werden. Im letzteren Fall spricht man vom sogenannten Service-Verfahren.

- *Studienstufe.* Der Zulassungsprozess zu einem Studiengang ist abhängig von der Studienstufe oder sogar innerhalb einer Stufe differenziert ausgestaltet. Während für die Zulassung zu einem Bachelorstudiengang lediglich die formale Hochschulzugangsberechtigung geprüft werden muss, sind auf Master- und Doktoratsstufe inhaltliche Prüfungen der bereits erworbenen Qualifikationen notwendig. Dabei ist es möglich, dass die vorhandenen Qualifikationen als nicht ausreichend beurteilt werden und eine Zulassung in den gewünschten Studiengang mit Auflagen versehen wird, d.h. bestimmte Qualifikationen aus einer tieferen Studienstufe innerhalb einer bestimmten Zeitspanne erworben werden müssen, um definitiv für den Studiengang zugelassen zu werden. Der Zulassungsprozess kann sich innerhalb der Studienstufen Master und Doktorat weiter differenzieren. Einerseits können Personen, die die Universität und das Studienfach beim Stufenübertritt nicht wechseln, einfach zugelassen werden, während bei allen anderen die Qualifikationen und formalen Zulassungsvoraussetzungen geprüft werden müssen. Andererseits gibt es verschiedene Ausgestaltungen der Master- und Doktoratsstufe. Neben dem konsekutiven Masterstudiengang, der an ein Bachelorstudium im selben Fach anschließt, gibt es auch spezialisierte Masterstudiengänge. Diese zeichnen sich dadurch aus, dass sie eine Vertiefung in einem interdisziplinären Themengebiet darstellen, d.h. insbesondere, dass sich Bachelorabsolventen verschiedener Fachrichtungen bewerben können und es keinen Bachelor-

studiengang mit derselben Ausrichtung gibt. Auf Doktoratsstufe gibt es drei Ausprägungen: das klassische Doktorat, Doktoratsprogramme und das fast track Doktorat. Letzteres steht besonders Begabten offen, die sich bereits mit dem Bachelorabschluß bewerben können. Sie müssen jedoch ihre Eignung unter Beweis stellen und sowohl Master- als auch Doktoratslehrveranstalungen besuchen.

- *Fachsemester.* Die Immatrikulation in einen Studiengang erfolgt in der Regel in das erste Fachsemester. Abhängig von der Studienordnung des Studiengangs kann sich eine Bewerberin bzw. ein Bewerber mit anrechenbaren Studienleistungen in ein höheres Fachsemester einschreiben lassen. Über die Anrechenbarkeit entscheidet die Fakultät bzw. der Fachbereich, der den Studiengang anbietet. Die Prüfung der erbrachten Leistungen kann aufgrund der Lehrveranstaltungsbezeichnung erfolgen, wenn diese an derselben Fakultät bzw. am selben Fachbereich erworben wurden. Umfangreichere Prüfungen werden notwendig, wenn die Leistungen von anderen Fakultäten oder Hochschulen stammen, da Inhalt und der Umfang der bereits abgelegten Prüfungen mit den Anforderungen des Studiengangs verglichen werden müssen.

- *Ort des Erwerbs der Hochschulzulassungsberechtigung.* Voraussetzung für die Immatrikulation an einer Hochschule ist die sogenannte Hochschulzulassungsberechtigung. Bei Bachelorstudiengängen ist dies in der Regel das Abitur. Bei höheren Studienstufen ist es in der Regel der erfolgreiche Abschluss der vorangegangenen Studienstufen. Im Zulassungsprozess wird die Hochschulzulassungsberechtigung formal geprüft, wobei Unterscheidungen aufgrund des Orts bzw. des Landes des Erwerbs der Hochschulzulassungsberechtigung vorgenommen werde. Wurde der Abschluss im Inland oder der Europäischen Union erworben, kann er ohne weitere Prüfungen akzeptiert werden. Stammt der Abschluss aus einem Land außerhalb der EU, ist die Äquivalenz zu einem inländischen Abschluss zu prüfen. Beim Schulabschluss wird beispielsweise geprüft, ob der Fächerkanon mit dem des Abiturs übereinstimmt.

Ausgangspunkt der folgenden Betrachtungen ist der Prozess zur Zulassung in das erste Semester eines Bachelorstudiengangs für eine Bildungsinländerin bzw. einen Bildungsinländer (siehe Abbildung 18). Für den Studiengang soll gelten, dass keine Zulassungsbeschränkungen erlassen wurden. Des Weiteren möchte die bzw. der Bewerbende keine etwaig bereits erbrachten Studienleistungen anrechnen lassen. Für alle Ausgabedatenobjekte des Prozesses gilt, dass sie im Studierendenverwaltungssystem abgelegt werden. Dieser Datenfluss ist aus Übersichtlichkeitsgründen nicht im Diagramm eingezeichnet. Das Studierendenverwaltungssystem ist eine externe Datenquelle bzw. -senke, d.h. sie steht auch anderen Prozessen zur Verfügung und enthält die Stammdaten der Studierenden sowie alle Informationen zu ihrem Studienverlauf.

4.2 Entstehen von Datenflussvarianten

Im Folgenden wird beschrieben, unter welchen Bedingungen Datenflussvarianten entstehen können. Dazu werden Änderungen bzw. Eigenschaften der fünf Workflow Perspektiven (vgl. Abschnitt Workflow Perspektiven) betrachtet. Tabelle 4 gibt einen Überblick über die Workflow Perspektiven und die vorgestellten Modifikationen.

4.2.1 Modifikation der Funktionsperspektive

Eine naheliegende Konstellation, in der sich der Datenfluss ändern kann, liegt vor, wenn die Aktivitäten eines Prozesses verändert werden. Werden Aktivitäten, die Daten als Eingabe benötigen oder

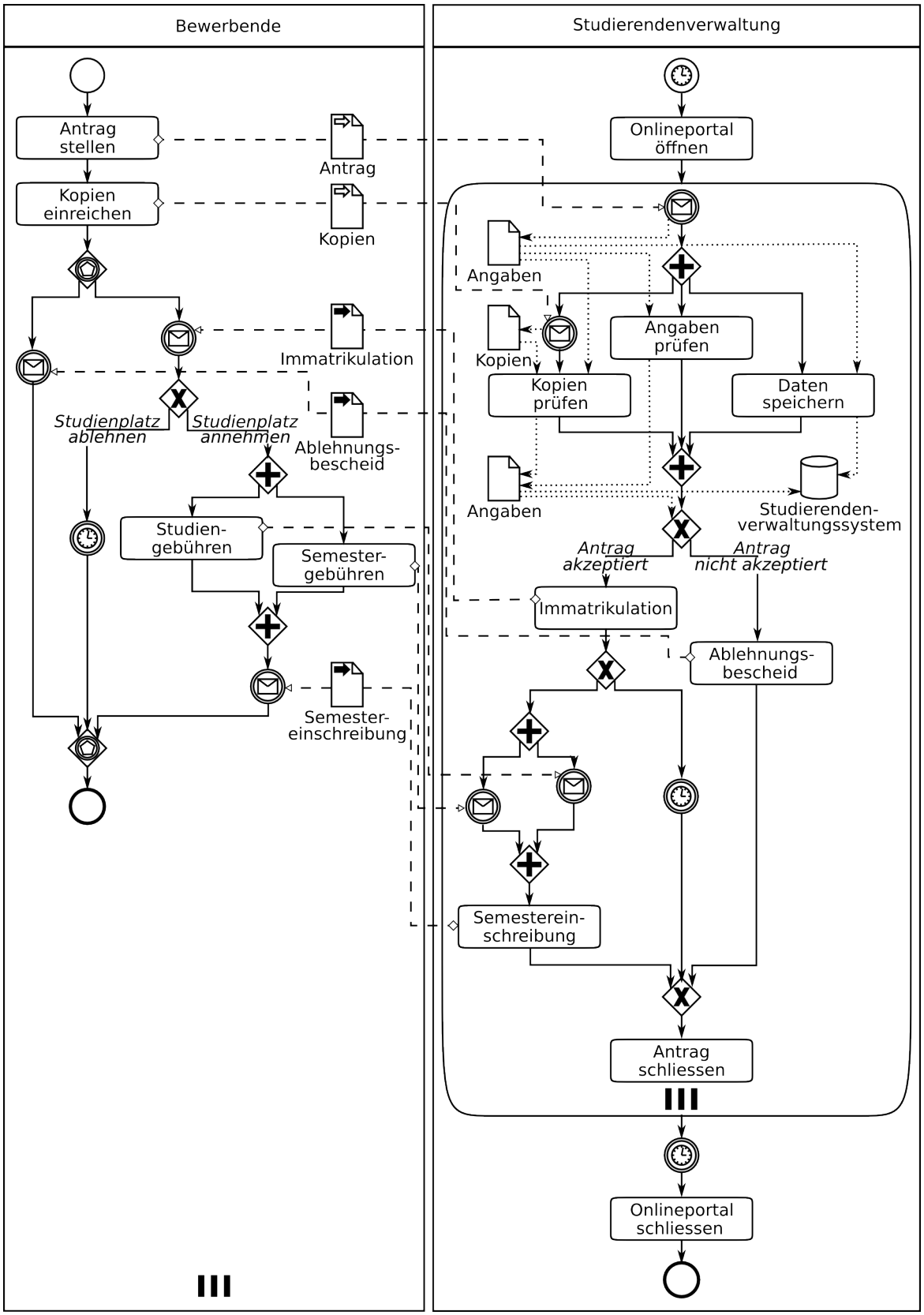


Abbildung 18: Zulassungsprozess für eine Bildungsinländerin bzw. einen Bildungsinländer in das erste Semester eines Bachelorstudiengangs

Workflow Perspektive	Art der Modifikation
Funktionsperspektive	Löschen einer Aktivität Einfügen einer Aktivität Modifizieren einer Aktivität
Verhaltensperspektive	Modifizieren der Reihenfolge der Aktivitäten Datenbasiertes Routing
Informationsperspektive	Löschen eines Datenobjekts Einfügen eines Datenobjekts Modifizieren eines Datenobjekts: • Visibilität • Datenstruktur/Granularität • Vertraulichkeit
Operationsperspektive	Modifizieren der Reihenfolge der Datenbearbeitung Übermittlungsart
Organisationsperspektive	Löschen einer Rolle Einfügen einer Rolle Modifizieren einer Rolle: • Kompetenzen/Rechte

Tabelle 4: Änderungen an den fünf Workflow Perspektiven, die zu Datenflussvarianten führen können.

Daten als Ergebnis generieren, gelöscht oder neu in einen Prozess eingefügt, muss der Datenfluss entsprechend angepasst werden. Gleiches gilt, wenn eine Aktivität modifiziert wird und sie davor bzw. danach Daten benötigt, d.h. mit dem Datenfluss verbunden ist [Rei00].

Löschen einer Aktivität Wird eine Aktivität eines Prozesses, die mit mindestens einem Datenobjekt verbunden ist, gelöscht, muss entschieden werden, was dies für den Datenfluss bedeutet. Wird durch das Löschen einer Aktivität und ihrer Modifikation des Datenobjekts der gesamte Datenfluss hinfällig, muss dieser aus dem gesamten Prozess entfernt werden. Ist die Modifikation, die die gelöschte Aktivität auf den Daten ausgeführt hat, obsolet, aber der Datenfluss nicht, muss der Datenfluss wieder hergestellt werden. Es ist jedoch auch möglich, dass nach dem Löschen der Aktivität eine Bearbeitung des Datenobjekts weiterhin notwendig ist, und dies nun durch eine andere im Prozess vorhandene Aktivität geschehen soll.

Erläuterung zum Löschen einer Aktivität anhand Abbildung 19. Die Aktivitäten A, B, C, D und E sowie das Datenobjekt Z sind Teil eines Geschäftsprozesses. Kontroll- und Datenfluss des Geschäftsprozesses sowie ihr Zusammenspiel sind korrekt. Zur Veranschaulichung wird, ohne Einschränkung der Gültigkeit für andere Anordnungen, ein sequenzieller Kontroll- und Datenfluss zwischen den Aktivitäten angenommen (vgl. Abbildung 19a). Der Teil des Datenflusses, der mit den erwähnten Aktivitäten verbunden ist, ist exemplarisch wie folgt gestaltet: Aktivität A liest das Datenobjekt Z und stellt es Aktivität B zur Verfügung. Aktivität B gibt das Datenobjekt in modifizierter Form an Aktivität D weiter.
Es soll gelten, dass die Korrektheit des Kontrollflusses nach dem Löschen von Aktivität B in einer

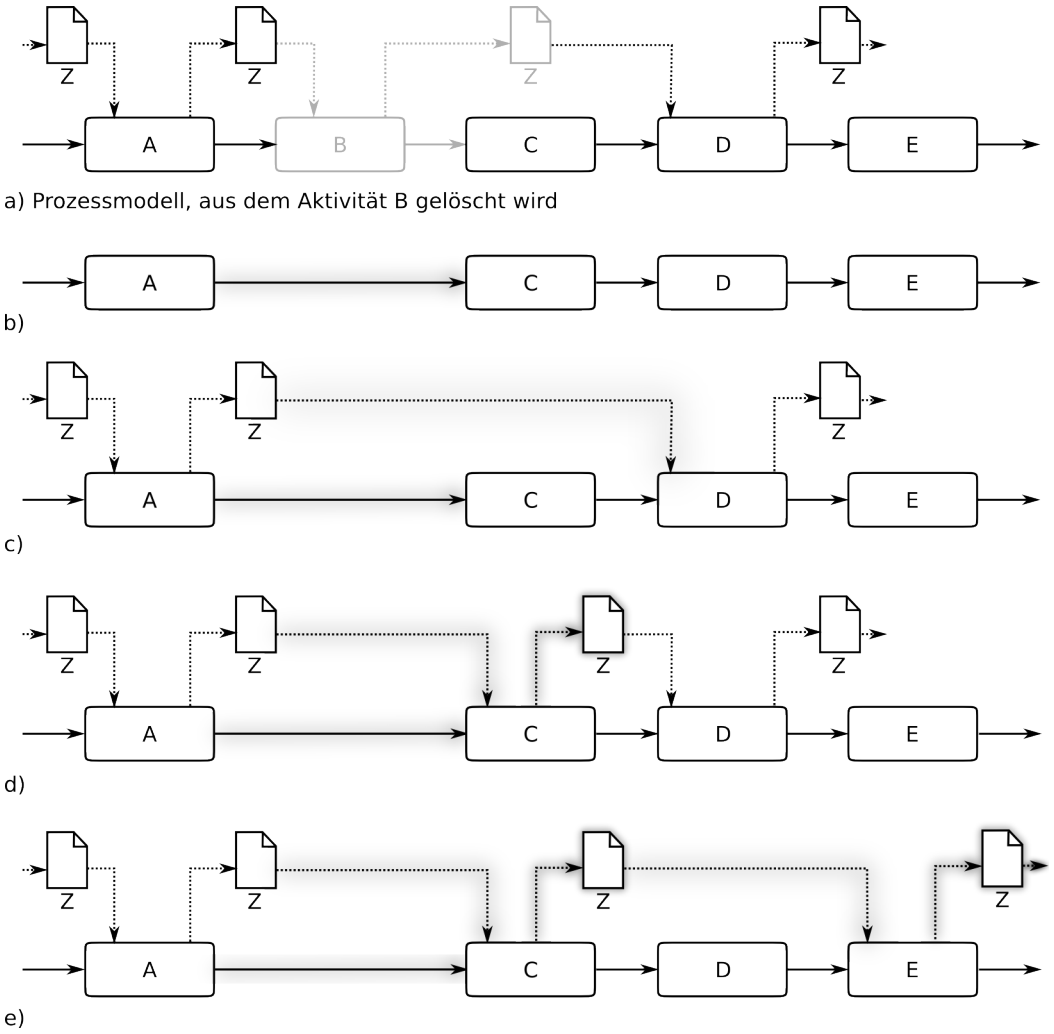


Abbildung 19: Löschen einer Aktivität aus einem Prozessmodell und mögliche Auswirkungen auf den Datenfluss

Prozessvariante weiterhin gegeben ist. Auf den Datenfluss kann das Löschen folgende Auswirkungen haben:

- Das Datenobjekt Z wird aus dem gesamten Prozess gelöscht, d.h. keine Aktivität des Prozesses erwartet das Datenobjekt als Eingabe, noch wird Z von einer Aktivität geschrieben. Darüber hinaus dient das Datenobjekt Z nicht für datenbasiertes Routing und wird weder von Kontextdaten initialisiert, noch dient es als Baustein um Kontextdaten zu erzeugen (vgl. Abbildung 19b).
- Das von Aktivität A geschriebene Datenobjekt wird direkt der Aktivität D zur Verfügung gestellt (vgl. Abbildung 19c).
- Das von A geschriebene Datenobjekt wird der bisher nicht in den Datenfluss eingebundenen Aktivität C zur Verfügung gestellt. Diese schreibt das Datenobjekt Z und stellt es
  - entweder der Aktivität D zur Verfügung und der von dort ausgehende bereits modellierte Datenfluss wird verfolgt. In diesem Fall substituiert Aktivität C die gelöschte Aktivität B (vgl. Abbildung 19c), oder stellt es
  - einer bisher nicht in den Datenfluss eingebundenen Aktivität E zur Bearbeitung zur Verfügung und es wird ein neuer von dieser Aktivität ausgehender Datenfluss für das Datenobjekt Z modelliert. Der Teil des Datenflusses, der ursprünglich von Aktivität B ausging, wird komplett gelöscht (vgl. Abbildung 19d).

  In beiden Fällen ist sicherzustellen, dass die Korrektheit des Datenflusses in Verbindung mit dem Kontrollfluss gewährleistet ist und das eventuell vorhandene datenbasierte Routing, sowie mögliche Kontextdaten berücksichtigt werden.
- Handelt es sich bei Aktivität B um eine mehrfach instanziierte Aktivität, die durch das Löschen zu einer einfach instanziierten Aktivität geändert wird, hat dies zur Folge, dass die entsprechenden Referenzdaten nicht mehr benötigt werden. Steht mit der Aktivität eine mehrfach beteiligten Rolle in Verbindung, wird diese zu einer einfach beteiligten Rolle geändert. Kontroll- und Datenfluss bleiben unverändert.

Beispiel 4.1. (Bewerbung auf Zulassung zu einem Studiengang mit bundesweiter Zulassungsbeschränkung) Ausgehend von dem in Abbildung 18 beschriebenen Prozess zur Zulassung in einen Bachelorstudiengang ohne Zulassungsbeschränkung, kann eine Variante für die bundesweit zulassungsbeschränkten Studiengänge beschrieben werden. Für diese gilt, dass die Bewerbenden ihre Unterlagen bei der Stiftung für Hochschulzulassung einreichen und die Stiftung über die Bewerbung entscheidet. Die Hochschule nimmt anschließend die bereits geprüften und gutgeheißenen Anträge von der Stiftung entgegen und verschickt die Immatrikulationsunterlagen. Um diese Variante zu modellieren, muss der Prozess in Abbildung 18 verschiedene Anpassungen erfahren, darunter auch, dass die Aktivität „Onlineportal öffnen“ und alle Aktivitäten der Studierendenverwaltung, die zur formalen Prüfung der Unterlagen dienen, gelöscht werden. Genauso wie der Entscheidungsknoten, der über den Versand der Immatrikulationsunterlagen oder des Ablehnungsbescheids entscheidet. Mit dem Entscheidungsknoten wird der Pfad für die Zustellung eines Ablehnungsbescheids gelöscht. Für die Datenobjekte des Prozesses bedeutet dies, dass der Prozess der Studierendenverwaltung durch Daten der Stiftung für Hochschulzulassung initiiert wird. Diese Daten umfassen

den Antrag einer Bewerberin bzw. eines Bewerbers einschließlich der beglaubigten Kopien und der Information, dass beides geprüft wurde und gutgeheißen wurde. Die Angaben zur Bewerbung werden, wie beim Prozess ohne Zulassungsbeschränkung, im Studierendenverwaltungssystem der Hochschule erfasst, d.h. insbesondere die Stammdaten der Person gespeichert. Das Löschen des Pfads mit dem Ablehnungsbescheid hat zur Folge, dass dieses Dokument bzw. diese Dokumentenvorlage in dieser Variante nicht zur Verfügung stehen muss. Absagen werden stattdessen von der Stiftung für Hochschulzulassung direkt verschickt.

Einfügen einer Aktivität Wird eine Aktivität in den Kontrollfluss eines Prozesses eingefügt, ist zu prüfen, ob und wie sie mit den bestehen Datenflüssen verbunden werden muss. Das Einfügen einer Aktivität kann darüber hinaus bedeuten, dass ein neues Datenobjekt erzeugt wird, d.h. ein neuer Datenfluss entsteht.

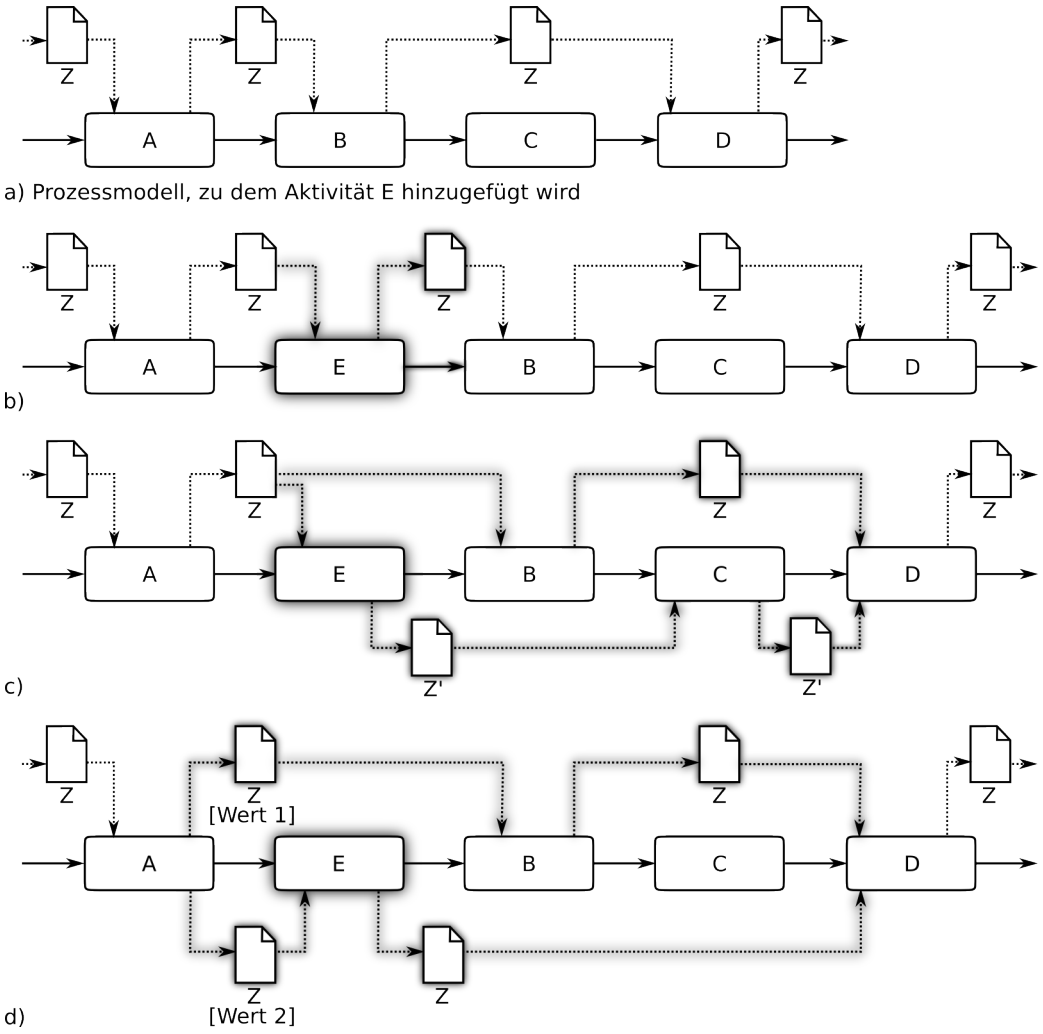


Abbildung 20: Einfügen einer Aktivität E in ein Prozessmodell und mögliche Auswirkungen auf den Datenfluss

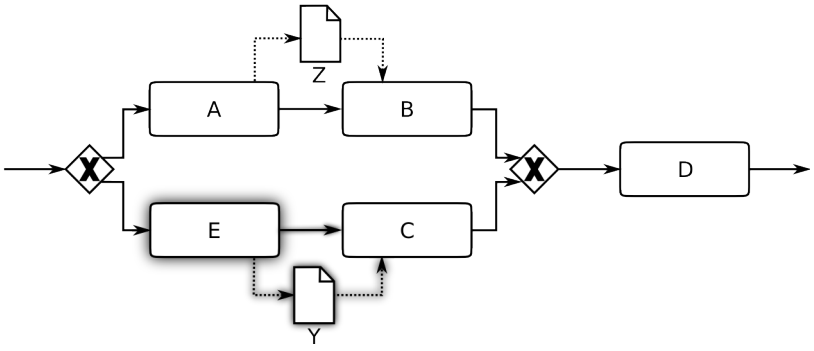


Abbildung 21: Einfügen einer Aktivität E mit einem Datenobjekt Y in ein Prozessmodell in Form eines alternativen Datenflusses

Erläuterung zum Einfügen einer Aktivität anhand Abbildungen 20 und 21. Die Aktivitäten A, B, C und D sowie das Datenobjekt Z sind Teil eines Geschäftsprozesses. Kontroll- und Datenfluss des Geschäftsprozesses sowie ihr Zusammenspiel sind korrekt. Zur Veranschaulichung wird, ohne Einschränkung der Gültigkeit für andere Anordnungen, ein sequenzieller Kontroll- und Datenfluss zwischen den Aktivitäten angenommen (vgl. Abbildung 20a). Der Teil des Datenflusses, der mit den erwähnten Aktivitäten verbunden ist, ist exemplarisch wie folgt gestaltet: Aktivität A liest das Datenobjekt Z und stellt es Aktivität B zur Verfügung. Aktivität B gibt das Datenobjekt an Aktivität D weiter.
Eine Aktivität kann sequenziell, parallel oder alternativ zum bisherigen Kontrollfluss eingefügt werden. Die Korrektheit des Kontrollflusses soll nach dem Einfügen der Aktivität E in den Prozess weiterhin gegeben sein. Auf den Datenfluss kann das Einfügen folgende Auswirkungen haben:

- Die Aktivität E kann durch einen sequenziellen, parallelen oder alternativen Datenfluss mit dem Datenobjekt Z versorgt werden. Im Folgenden wird davon abstrahiert, dass nicht alle Kontroll- und Datenflusskombinationen sinnvoll sind und nur auf die Datenflussarten fokussiert. Dabei wird davon ausgegangen, dass die Aktivität E entweder sequenziell zwischen die Aktivitäten A und B eingefügt wird, oder sie sich auf einem parallelen bzw. alternativen Kontrollflusspfad zur Aktivität B befindet. In Abbildung 20 wird beispielhaft das sequenzielle Einfügen veranschaulicht.

  - *Sequenzieller Datenfluss.* Das Datenobjekt Z wird von Aktivität A an Aktivität E weitergegeben. Anschließend liest Aktivität B das Datenobjekt und stellt es der Aktivität D zur Verfügung (vgl. Abbildung 20b).
  - *Paralleler Datenfluss.* Aktivität A schreibt das Datenobjekt Z. Dieses wird an die Aktivitäten E und B parallel weitergeben. Ausgehend von E wird ein neuer paralleler Datenfluss modelliert, der exemplarisch wie folgt gestaltet sein könnte: Aktivität E schreibt ein neues Datenobjekt Z', da das Datenobjekt Z nicht überschrieben werden darf. Z' wird an Aktivität C weitergegeben. Diese gibt es anschließend an Aktivität D weiter. Gleichzeitig wird der bereits modellierte Datenfluss des Datenobjekts Z verfolgt, d.h. Aktivität B gibt das Datenobjekt Z an Aktivität D weiter (vgl. Abbildung 20c). In der Aktivität, in der sich die parallelen Datenflüsse treffen, in diesem Beispiel in Aktivität D, ist zu prüfen, ob mit beiden Datenobjekten weitergearbeitet werden soll, oder ob der Datenfluss wieder vereinigt werden kann.

  - *Alternativer Datenfluss von Werten.* Die Aktivität A gibt das Datenobjekt Z entweder mit dem Wert 1 oder den Wert 2 weiter. Wird Wert 1 geschrieben, wird der bereits modellierte Datenfluss verfolgt, d.h. das Datenobjekt an Aktivität B und anschließend an Aktivität D weitergegeben. Hat das Datenobjekt Z den Wert 2 nimmt es einen alternativen Weg. In diesem Fall wird das Datenobjekt der Aktivität E zur Verfügung gestellt, von dieser verarbeitet und anschließend an Aktivität D weitergegeben (vgl. Abbildung 20d). Die Aktivität, in der die alternativen Datenflüsse münden, in diesem Beispiel Aktivität D, benötigt das Datenobjekt Z unabhängig von dem eingeschlagenen Datenfluss.

- Das Einfügen der Aktivität E in den Prozess kann damit verbunden sein, dass ein neues, bisher nicht im Prozess vorhandenes Datenobjekt Y eingefügt wird. Für dieses Datenobjekt ist ein neuer Datenfluss zu modellieren. Der bestehende Datenfluss für das Datenobjekt Z wird dadurch nicht tangiert. Eine erwähnenswerte Konstellation, die entstehen kann, ist der *alternative Datenfluss von Datenobjekten.* Dieser liegt vor, wenn das vorgestellte Szenario dahingehend modifiziert wird, dass Datenobjekt Z von Aktivität A erstellt wird, d.h. der Datenfluss in dieser Aktivität beginnt und sich die neu eingefügte Aktivität E auf einem alternativen Kontrollflusspfad zu Aktivität A befindet und die Aktivität E ein Datenobjekt Y erstellt (vgl. Abbildung 21). Alternative Datenflüsse können auf zwei verschiedene Arten enden: Entweder jeweils innerhalb der alternativen Kontrollflusspfade oder in einer Aktivität nach dem Zusammenführen der alternativen Kontrollflusspfade, wobei diese Aktivität so ausgelegt sein muss, dass sie unabhängig davon, welches Datenobjekt sie erhält, ausgeführt werden kann.

- Wird eine Aktivität zu einer mehrfach instanziierten Aktivität, hat dies zur Folge, dass Referenzdaten benötigt werden. Steht mit der Aktivität eine Rolle in Verbindung, wird diese zu einer mehrfach beteiligten Rolle geändert.

Beispiel 4.2. (Bewerbung auf Zulassung in ein höheres Fachsemester eines Studiengangs) Möchte sich eine Bewerberin bzw. ein Bewerber in ein höheres Fachsemester eines Studiengangs einschreiben, ist der allgemeine Prozess, der in Abbildung 18 gezeigt wird, nicht ausreichend. Neben der Tatsache, dass zusätzliche Unterlagen bei der Studierendenverwaltung eingereicht werden müssen, die über die bereits erworbenen Leistungen Auskunft geben, sind zusätzliche Aktivitäten notwendig, um die Unterlagen entgegen zu nehmen und zu verarbeiten. So nimmt die Studierendenverwaltung die formale Zulassungsprüfung vor und gibt die Unterlagen über die bereits erbrachten Leistungen an die zuständige Fakultät weiter. Diese nimmt deren inhaltliche Prüfung vor und teilt der Studierendenverwaltung das Ergebnis mit. Aufgrund der formalen und der fachlichen Prüfung kann darüber entschieden werden, ob und wenn ja, in welches Fachsemester eine Zulassung erfolgen kann. Die Studierendenverwaltung teilt dies der Bewerberin bzw. dem Bewerber in einem entsprechenden Bescheid mit.

Modifizieren einer Aktivität Die Modifikation einer Aktivität kann bedeuten, dass ein Datenobjekt, das ursprünglich mit dieser Aktivität verbunden war, nun nicht mehr von ihr bearbeitet wird. In diesem Fall treffen dieselben Überlegungen wie beim Löschen einer Aktivität zu. Es kann aber auch bedeuten, dass die Aktivität nun ein Datenobjekt benötigt, das zuvor nicht von ihr bearbeitet wurde. Bei diesem Fall treffen dieselben Überlegungen wie beim Einfügen einer Aktivität zu.

4.2.2 Modifizieren der Verhaltensperspektive

Änderung der Anordnung der Aktivitäten Varianten eines Prozesses können sich in der Anordnung der Aktivitäten unterscheiden. Die sequenzielle Reihenfolge der Aktivitäten kann verändert werden, aber auch die Kontrollflussart kann modifiziert werden, d.h. eine Sequenz kann durch einen parallelen oder alternativen Kontrollfluss ersetzt werden und umgekehrt. Sind Datenobjekte mit diesen Aktivitäten verbunden, ist sicherzustellen, dass der Datenfluss weiterhin korrekt ist.

Erläuterung zur Änderung der Anordnung der Aktivitäten anhand Abbildung 22. Die Aktivitäten A, B, C und D sowie das Datenobjekt Z sind Teil eines Geschäftsprozesses. Kontroll- und Datenfluss des Geschäftsprozesses sowie ihr Zusammenspiel sind korrekt. Zur Veranschaulichung wird, ohne Einschränkung der Gültigkeit für andere Anordnungen, ein sequenzieller Kontroll- und Datenfluss zwischen den Aktivitäten angenommen (vgl. Abbildung 22a). Dabei fließt das Datenobjekt Z von A über B und C nach D. Die Anordnung der Aktivitäten kann wie folgt modifiziert werden:

- Wird wie in Abbildung 22b) zu sehen die Reihenfolge der Aktivitäten A und B vertauscht und der Datenfluss beibehalten, erwartet die nun zuerst ausgeführte Aktivität B ein von der nachfolgenden Aktivität A zur Verfügung gestelltes Datenobjekt. Dies führt zu einer Verklemmung. Um diese zu vermeiden, könnte beispielsweise die Reihenfolge der Datenbearbeitung an die des neuen Kontrollflusses angepasst werden.

- Wie bereits beim Einfügen einer Aktivität angedeutet, sind nicht alle Kontroll- und Datenflusskombinationen gleich sinnvoll. Wird die Kontrollflussart geändert, ist zu untersuchen, ob der Datenfluss in der vorliegenden Art und Weise weiterbestehen kann. In Abbildung 22c) wird exemplarisch angenommen, dass die Aktivitäten B und C neu parallel zu A und D ausgeführt werden und der ursprüngliche Datenfluss bestehen bleibt. Kontroll- und Datenfluss dieses Prozessmodells sind zwar korrekt, die Ausführung von Aktivität D ist aber von dem von Aktivität C bearbeiteten Datenobjekt Z abhängig, so dass Aktivität D nicht parallel, sondern weiterhin nach Aktivität C ausgeführt wird. Die gleiche Abhängigkeit besteht zudem zwischen den Aktivitäten A und B.
  Neben diesem Szenario gibt aber auch Konstellationen, die sich ausschließen. So ist es nicht möglich das eingangs erwähnte Beispiel mit sequenziellen Kontroll- und Datenfluss zu einem alternativen Kontrollfluss unter Beibehaltung des Datenflusses zu modellieren. Wie in Abbildung 22d) illustriert, würde dies für den einen Kontrollflusspfad bedeuten, dass Aktivität A ausgeführt wird und dabei das Datenobjekt Z schreibt. Im Anschluss daran wartet jedoch Aktivität D vergeblich, dass Aktivität C das Datenobjekt Z bereitstellt. Wird der andere Kontrollflusspfad ausgewählt, wartet bereits die erste auszuführende Aktivität B vergeblich auf das von Aktivität A modifizierte Datenobjekt Z.

Beispiel 4.3. (Bewerbung auf Zulassung zu einem Studiengang an einer Hochschule) Der in Abbildung 18 beschriebene Prozess sieht vor, dass der Online-Antrag der Bewerberin bzw. des Bewerbers von der Studierendenverwaltung entgegengenommen wird. Anschließend werden die Daten gleichzeitig weitergegeben, um die beglaubigten Kopien zu ergänzen, formal zu prüfen sowie im Studierendenverwaltungssystem abzulegen. Um dies zu realisieren, liegt sowohl ein paralleler Kontroll- als auch Datenfluss vor. In einer Prozessvariante könnte nun vorgesehen werden, die Informationen erst in der Datenbank abzulegen, wenn die formale Prüfung abgeschlossen wurde, d.h. der parallele Kontrollfluss wird für diese beiden Äste zu einem sequenziellen Kontrollfluss umge-

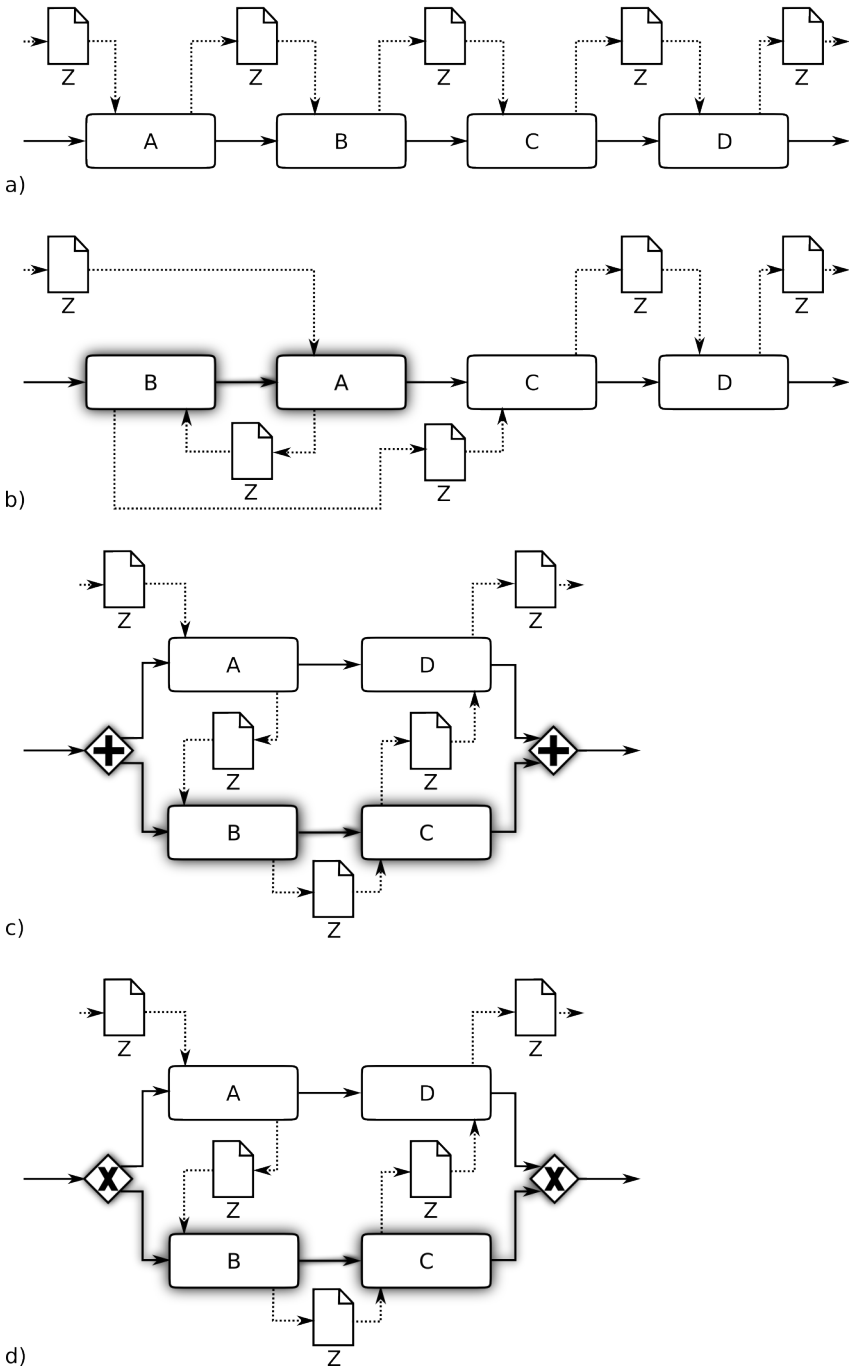


Abbildung 22: Modifizieren der Anordnung der Aktivitäten eines Prozessmodells und mögliche Auswirkungen auf den Datenfluss

formt. Der Datenfluss wird ebenfalls modifiziert und fließt sequenziell von der Aktivität, die sie prüft zu der, die sie in der Datenbank ablegt.

Datenbasiertes Routing Datenbasiertes Routing liegt vor, wenn bei der Ausführung des Prozesses an `OR`- oder `XOR`-Verzweigungsknoten aufgrund der Werte eines oder mehrerer Datenobjekte entschieden wird, welche ausgehende Kontrollflusskante weiterverfolgt wird. Die Verzweigungsknoten werden zu diesem Zweck mit den entsprechenden Datenobjekten aus dem Prozess versorgt. Dies wird, wie bereits beschrieben, durch in die Verzweigungsknoten eingehende Datenkanten verdeutlicht. Wird ein Verzweigungsknoten gelöscht bzw. so modifiziert, dass das ursprüngliche Datenobjekt nicht mehr für die Ablaufentscheidung benötigt wird, so wird auf die zum Verzweigungsknoten führende Datenkante verzichtet. Der Datenfluss des Datenobjekts bleibt ansonsten unverändert. Wird ein Verzweigungsknoten neu in einen Prozess eingefügt, oder wird ein zusätzliches Datenobjekt benötigt, um die Ablaufentscheidung treffen zu können, ist zu prüfen, ob das entsprechende Datenobjekt im Prozess vorliegt und nach welchem Bearbeitungsschritt es mit dem Verzweigungsknoten lesend verbunden werden soll. Sind die entsprechenden Daten nicht im Prozess vorhanden, ist ein entsprechendes Datenobjekt in den Prozess einzufügen.

4.2.3 Modifizieren der Informationsperspektive

Änderungen an den Daten eines Prozesses bzw. ihren Eigenschaften können ebenfalls zu Datenflussvarianten führen. Offensichtlich wird der Datenfluss modifiziert, wenn Datenobjekte aus einer Prozessvarianten gänzlich gelöscht oder neu hinzugefügt werden. Jedoch auch durch die Modifikation der Visibität, Granularität und Vertraulichkeit eines Datenobjekts können neue Datenflussvarianten entstehen.

Löschen eines Datenobjekts In einer Prozessvariante kann ein Datenobjekt vollständig fehlen, d.h. aus dem gesamten Prozess gelöscht werden, oder es kann punktuell aus dem Prozess gelöscht werden, etwa wenn eine einzelne Aktivität, die das Datenobjekt ursprünglich verarbeitet hat, diese Informationen nicht mehr benötigt. In diesem Fall ist zu entscheiden, wie der Datenfluss wieder korrekt hergestellt werden kann.

Erläuterung zum Löschen eines Datenobjekts anhand Abbildung 23. Die Aktivitäten A, B, C, D und E sowie das Datenobjekt Z sind Teil eines Geschäftsprozesses. Kontroll- und Datenfluss des Geschäftsprozesses sowie ihr Zusammenspiel sind korrekt. Zur Veranschaulichung wird, ohne Einschränkung der Gültigkeit für andere Anordnungen, ein sequenzieller Kontroll- und Datenfluss zwischen den Aktivitäten angenommen (vgl. Abbildung 23a). Der Teil des Datenflusses, der mit den erwähnten Aktivitäten verbunden ist, ist exemplarisch wie folgt gestaltet: Aktivität A liest das Datenobjekt Z und stellt es Aktivität B zur Verfügung. Aktivität B gibt das Datenobjekt an Aktivität D weiter.
Das Löschen des Datenobjekts Z kann vollständig oder punktuell erfolgen:

- Das Datenobjekt Z wird aus dem gesamten Prozess gelöscht, d.h. keine Aktivität des Prozesses erwartet das Datenobjekt Z als Eingabe, noch wird es von einer Aktivität geschrieben. Darüber hinaus dient das Datenobjekt Z nicht für datenbasiertes Routing und wird weder von Kontextdaten initialisiert, noch dient es als Baustein, um Kontextdaten zu erzeugen (vgl. Abbildung 23b).

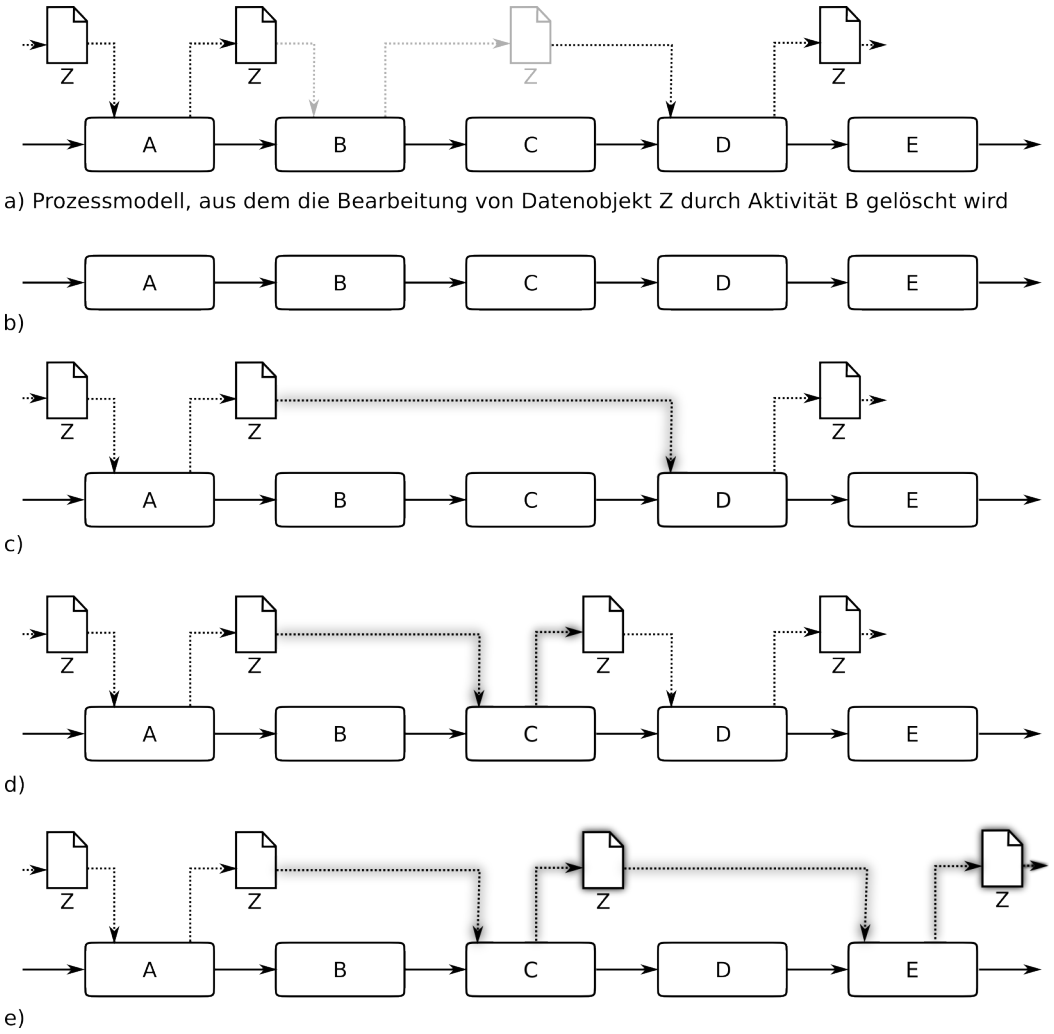


Abbildung 23: Löschen eines Datenobjekts aus einem Prozessmodell und mögliche Auswirkungen auf den Datenfluss

- Aktivität B soll das Datenobjekt Z nicht mehr verarbeiten. Dies kann folgende Auswirkungen haben:
  - Das von Aktivität A geschriebene Datenobjekt Z wird der Aktivität D direkt zur Verfügung gestellt (vgl. Abbildung 23c).
  - Das von A geschriebene Datenobjekt wird einer anderen im Prozess vorhandenen Aktivität C zur Verfügung gestellt. Diese schreibt das Datenobjekt und stellt es entweder
    * der Aktivität D zur Verfügung und der von dort ausgehende bereits modellierte Datenfluss wird verfolgt. In diesem Fall substituiert Aktivität C die Modifikation des Datenobjekts durch Aktivität B (vgl. Abbildung 23d).
    * oder das Datenobjekt Z wird der anderen im Prozess vorhandenen Aktivität E zur Bearbeitung zur Verfügung gestellt und ein von dieser Aktivität ausgehender Datenfluss für das Datenobjekt Z modelliert. Der Teil des Datenflusses, der ursprünglich von Aktivität D ausging, wird gelöscht (vgl. Abbildung 23e).

  In beiden Fällen ist sicherzustellen, dass die Korrektheit des Datenflusses in Verbindung mit dem Kontrollfluss gewährleistet ist und das eventuell vorhandene datenbasierte Routing sowie mögliche Kontextdaten berücksichtigt werden.

Beispiel 4.4. (Bewerbende mit ausländischer Vorbildung) Eine Variante für den Prozess zur Zulassung in einen Studiengang ist für Bewerbende mit ausländischer Vorbildung vorgesehen. Dabei gliedert sich diese in weitere Varianten für eine Vorbildung aus der Europäischen Union und eine Vorbildung aus dem übrigen Ausland. Prinzipiell können diese Varianten vorsehen, dass ein Nachweis genügender Deutschkenntnisse in Form einer Deutschprüfung erbracht wird. Ist die Muttersprache der Bewerberin bzw. des Bewerbers jedoch Deutsch oder stammt ihre bzw. seine Vorbildung aus Österreich oder der Deutschschweiz, wird die Deutschprüfung erlassen. Dies bedeutet, dass in diesen Prozessvarianten das Datenobjekt Deutschprüfung aus dem gesamten Prozess gelöscht wird.

Einfügen eines Datenobjekts In einer Prozessvariante kann gegenüber dem ursprünglichen Prozessmodell ein neues Datenobjekt oder ein neuer Verarbeitungsschritt für ein bereits vorhandenes Datenobjekt benötigt werden. Im ersten Szenario ist ein neuer Datenfluss in den Prozess einzufügen, im zweiten sind eine oder mehrere Aktivitäten neu in den bestehenden Datenfluss einzubeziehen.

Erläuterung zum Einfügen eines Datenobjekts anhand Abbildung 24. Die Aktivitäten A, B, C und D sowie das Datenobjekt Z sind Teil eines Geschäftsprozesses. Kontroll- und Datenfluss des Geschäftsprozesses sowie ihr Zusammenspiel sind korrekt. Zur Veranschaulichung wird, ohne Einschränkung der Gültigkeit für andere Anordnungen, ein sequenzieller Kontroll- und Datenfluss zwischen den Aktivitäten angenommen (vgl. Abbildung 24a). Der Teil des Datenflusses, der mit den erwähnten Aktivitäten verbunden ist, ist exemplarisch wie folgt gestaltet: Aktivität A liest das Datenobjekt Z und stellt es Aktivität B zur Verfügung. Aktivität B gibt das Datenobjekt an Aktivität D weiter.

- Es kann ein neues Datenobjekt Y in den Prozess eingefügt werden. Für dieses Datenobjekt ist ein neuer Datenfluss zu modellieren. Der bestehende Datenfluss des Datenobjekts Z wird dadurch nicht tangiert (vgl. Abbildung 24b). Eine erwähnenswerte Konstellation, die entstehen

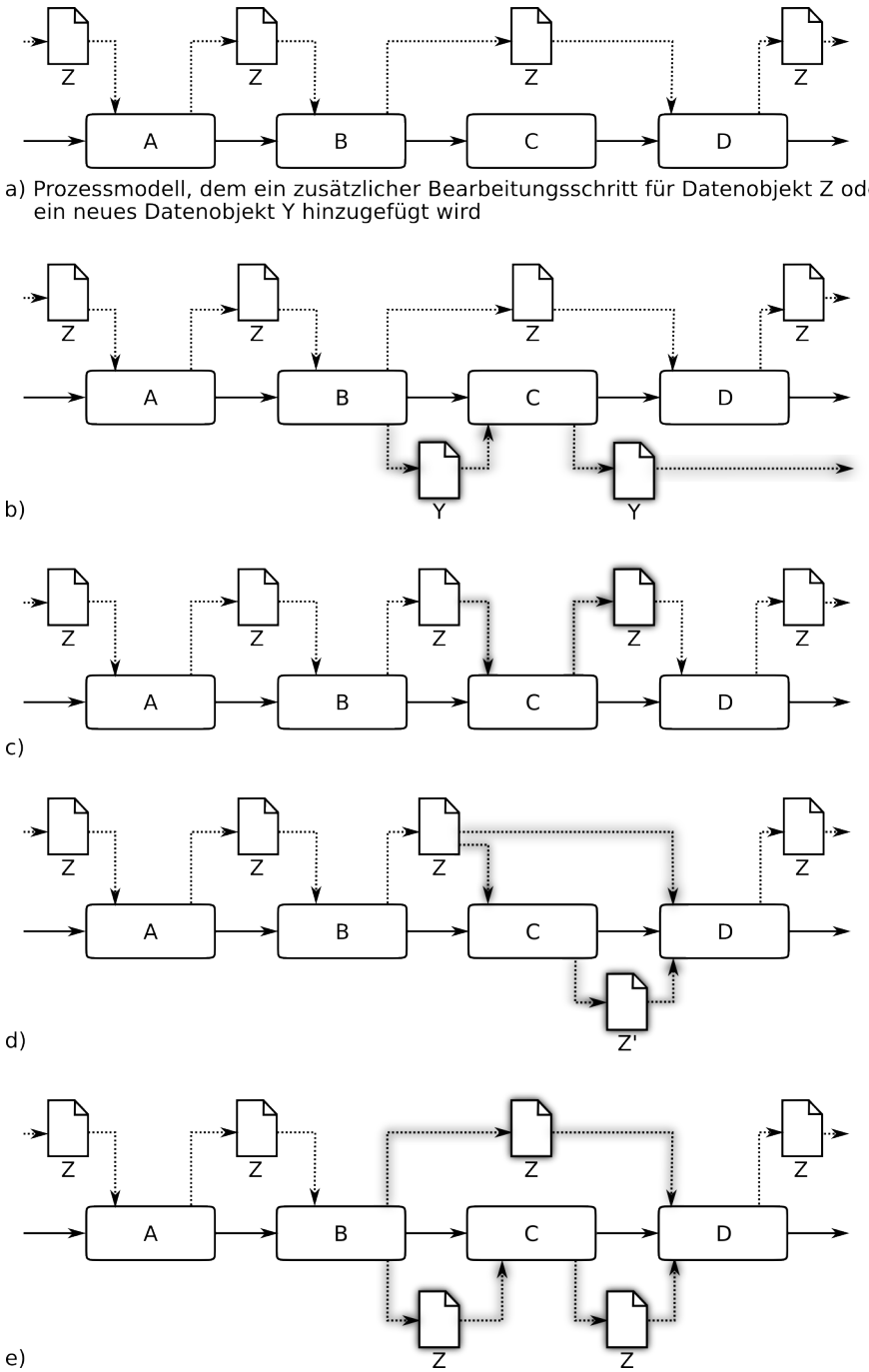


Abbildung 24: Einfügen eines Datenobjekts Y bzw. eines zusätzlichen Bearbeitungsschritts für Datenobjekt Z in ein Prozessmodell

kann, ist der *alternative Datenfluss von Datenobjekten*, d.h. die Datenflüsse der Datenobjekte Z und Y befinden sich auf alternativen Kontrollflusspfaden (vgl. Abbildung 21).

- Das Datenobjekt Z kann neu mit der bereits vorhandenen Aktivität C des Prozesses verbunden werden. Der Datenfluss zur Aktivität C kann sequenziell (vgl. Abbildung 24c), parallel (vgl. Abbildung 24d) oder als alternativer Datenfluss von Werten (vgl. Abbildung 24e) gestaltet werden.

Beispiel 4.5. (Studiengänge mit weiteren Zulassungsbedingungen) Für hoch spezialisierte Studiengänge kann eine Hochschule neben dem Abitur die Erfüllung weiterer Zulassungsbedingungen fordern. Typischerweise wird bei praxisorientierten Studiengängen das erfolgreiche Absolvieren eines mehrmonatigen Berufspraktikums verlangt. Diese Anforderung kann in einer Prozessvariante abgebildet werden, indem der Nachweis des Praktikums als zusätzliches Datenobjekt hinzugefügt wird.

Modifizieren von Datenobjekten Die Eigenschaften der Datenobjekte haben einen wesentlichen Einfluss darauf, ob Datenflussvarianten entstehen können. Dies soll anhand der Visibilität, der Datenstruktur und der Vertraulichkeit der Daten gezeigt werden.

Visibilität Ausgehend von den Betrachtungen in [RHEA04] zur Visibilität von Daten in Workflows, soll im Folgenden beleuchtet werden, welchen Einfluss die Visibilität auf den Datenfluss in einem Prozess haben kann. Datenobjekte können für eine Aktivität *(Task Data)*, eine Aktivität und ihre Subprozesse *(Block Data)*, einen festgelegten Bereich von Aktivitäten *(Scope Data)* oder für mehrere Instanzen einer Aktivität *(Multiple Instance Data)* sichtbar sein. Um einen Datenfluss außerhalb dieser sichtbaren Bereiche zu beschreiben, müssen die Daten entlang eines separat definierten Datenkanals weitergegeben werden. Anders verhält es sich mit Instanz- *(Case Data)*, Prozess- *(Workflow Data)* und globalen Daten *(Environment Data)*. Diese stehen allen Aktivitäten einer Instanz zur Verfügung und müssen nicht über Datenkanäle zwischen ihnen ausgetauscht werden. Wird die Visibilität eines Datenobjekts verändert, z.B. die auf einzelne Aktivitäten eingeschränkte Sichtbarkeit auf die gesamte Instanz erhöht, muss das Datenobjekt nicht mehr zwischen den Aktivitäten ausgetauscht werden, sondern kann von jeder Aktivität direkt aus dem Datenspeicher der Instanz gelesen und wieder zurück geschrieben werden.

Beispiel 4.6. (Studierendendaten) In dem Prozess zur Zulassung zu einem Studiengang an einer Hochschule besitzen die Datenobjekte unterschiedliche Visibilität. Das leere Antragsformular steht allen Instanzen des Prozesses zur Verfügung (Prozessdaten), während das ausgefüllte Antragsformular für jede Instanz individuell ist und von Aktivität zu Aktivität weitergegeben wird (Aktivitätsdaten). Die Studierendenstammdaten, die in der Datenbank abgelegt werden, sind globale Daten, auf die auch andere Prozesse zugreifen dürfen. Wird in einer Prozessvariante das ausgefüllte Antragsformular im Speicher der Instanz abgelegt (Instanzdaten), wird es für alle Aktivitäten der Instanz sichtbar und muss nicht mehr von Aktivität zu Aktivität weitergegeben werden.

Datenstruktur/Granularität Das Verändern der Datenstruktur kann unterschiedlich starke Auswirkungen auf einen Prozess haben. Werden leichte Veränderungen vorgenommen, etwa wenn eine Information nicht mehr mittels eines semistrukturierten, sondern eines strukturierten Datenobjekts

beschrieben werden soll, müssen die Verarbeitungsmechanismen der Aktivitäten des Prozesses angepasst werden, der Datenfluss bleibt aber unverändert. Anders verhält es sich, wenn mit der Änderung der Datenstruktur eine Veränderung der Granularität der Informationen einher geht, d.h. zwischen atomarer und komplexer Datenstruktur gewechselt wird und dies zu einer Zusammenfassung bzw. einer Trennung von Datenflüssen führt.

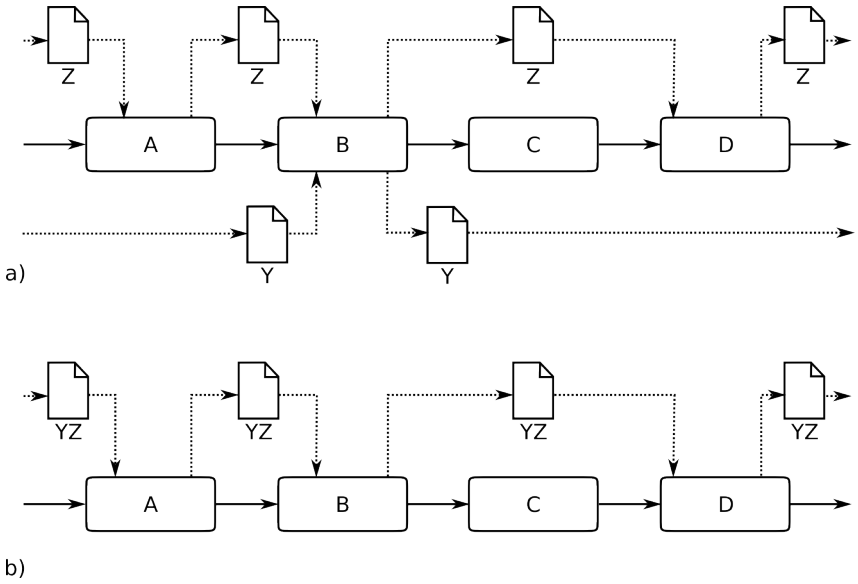


Abbildung 25: Modifizieren der Datenstruktur von Datenobjekten eines Prozessmodells

Erläuterung zur Datenstruktur/Granularität anhand Abbildung 25. Die Aktivitäten A, B, C und D sowie die Datenobjekte Y und Z sind Teil eines Geschäftsprozesses. Kontroll- und Datenfluss des Geschäftsprozesses sowie ihr Zusammenspiel sind korrekt. Zur Veranschaulichung wird, ohne Einschränkung der Gültigkeit für andere Anordnungen, ein sequenzieller Kontroll- und Datenfluss zwischen den Aktivitäten angenommen. Der Teil des Datenflusses, der mit den erwähnten Aktivitäten verbunden ist, ist exemplarisch wie folgt gestaltet: Aktivität A liest das Datenobjekt Z und stellt es Aktivität B zur Verfügung. Aktivität B gibt das Datenobjekt an Aktivität D weiter. Das Datenobjekt Y wird lediglich von Aktivität B gelesen und verarbeitet (vgl. Abbildung 25a). Ohne den Kontrollfluss zu ändern, werden die Datenobjekte Y und Z in einer Variante des Prozesses in einem komplexen Datenobjekt YZ vereint und der Datenfluss von YZ so gestaltet, dass alle Aktivitäten, die ursprünglich auf die Datenobjekte Y und/oder Z zugreifen mussten, berücksichtigt werden (vgl. Abbildung 25b).

Beispiel 4.7. (Modellierung der Bewerbungsunterlagen) Bei einem Zulassungsantrag in ein höheres Fachsemester eines Studiengangs werden von der Bewerberin bzw. dem Bewerber neben den üblichen Dokumenten wie Abiturzeugnis und Krankenversicherungsnachweis auch Unterlagen zu den bereits erbrachten Leistungen eingereicht. Alle eingereichten Unterlagen können einzeln für sich oder als gemeinsames Dossier betrachtet werden. Je nach dem wie dies gehandhabt wird, sind für jedes Dokument ein Datenfluss oder ein gemeinsamer Datenfluss für das Dossier zu modellieren.

Vertraulichkeit Eine wichtige Eigenschaft für den Austausch von Daten zwischen den an einem Prozess beteiligten Rollen ist die auf den Daten festgelegte Vertraulichkeitsstufe. Diese muss mit den Rechten einer Rolle (siehe Absatz Zugriffsrechte einer Rolle) übereinstimmen, um Zugriff auf die Daten zu ermöglichen. Prinzipiell können beliebig viele Vertraulichkeitsstufen festgelegt werden. Die Erhöhung der Vertraulichkeitsstufe geht mit einer Einschränkung der Adressatengruppe einher. Wird in einer Variante des Prozesses die Vertraulichkeitsstufe eines Datenobjekts herabgesetzt, muss dies keine Auswirkungen auf den bestehenden Datenfluss des Objekts haben, da die involvierten Rollen weiterhin zugriffsberechtigt sind. Es kann jedoch dazu führen, dass weitere Rollen, die bisher zu geringe Rechte besaßen, nun in den Datenfluss eingebunden werden – entweder indem der Datenfluss um diese Rollen erweitert wird, z.B. im Sinne einer Information, oder in der Form, dass Aufgaben an diese Personen delegiert werden und die ursprünglichen Rollen nur noch punktuell einbezogen werden. Das Einschränken der Vertraulichkeitsstufe eines Datenobjekts hat zwingend Auswirkungen auf den Datenfluss einer Prozessvariante, wenn man davon ausgeht, dass die beteiligten Rollen nur über die Zugriffsrechte verfügen, die für den ursprünglichen Prozess notwendig sind. Sollen in einer Prozessvariante dieselben Aktivitäten wie im ursprünglichen Prozess ausgeführt werden, kann dies nur geschehen, wenn sie an Rollen mit den entsprechenden Zugriffsrechten übergeben werden.

Erläuterung zur Vertraulichkeit anhand Abbildung 26. Die Aktivitäten A, B, C und D, die Rollen M und N sowie das Datenobjekt Z sind Teil eines Geschäftsprozesses. Kontroll- und Datenfluss des Geschäftsprozesses sowie ihr Zusammenspiel sind korrekt. Zur Veranschaulichung wird, ohne Einschränkung der Gültigkeit für andere Anordnungen, ein sequenzieller Kontroll- und Datenfluss zwischen den Aktivitäten angenommen (vgl. Abbildung 26a). Das Datenobjekt Z wird zwischen den Rollen M und N ausgetauscht. Datenobjekte werden in drei Vertraulichkeitsstufen eingeteilt: öffentlich, vertraulich und streng vertraulich. Die Rolle M hat das Zugriffsrecht auf alle Daten, einschließlich streng vertraulicher. N hat Zugriffsrechte, die vertrauliche Daten umfassen. Das Datenobjekt Z ist vertraulich.

- Wird das Datenobjekt Z in einer Prozessvariante als öffentlich deklariert, kann der Prozess entweder
  - wie bisher ablaufen, d.h. Kontroll- und Datenfluss bleiben unverändert oder
  - es wird eine neue Rolle O mit Zugriffsrecht auf öffentliche Daten in den Prozess einbezogen, an die verschiedene Aktivitäten delegiert werden können (vgl. Abbildung 26b).
- Wird das Datenobjekt Z mit streng vertraulich gekennzeichnet, darf N nicht mehr darauf zugreifen. Dies kann zur Folge haben, dass die entsprechenden Aktivitäten von der Rolle M ausgeführt werden müssen (vgl. Abbildung 26c), oder dass diese von einer neuen Rolle P mit Zugriffsrecht auf streng vertrauliche Daten übernommen werden (vgl. Abbildung 26d).

Beispiel 4.8. (Vertraulichkeit von Professorendaten) Personenbezogene Daten werden prinzipiell vertraulich gehandhabt. Bei den Daten von Professuren gibt es jedoch Ausnahmen. Beispielsweise ist die berufliche Anschrift einschließlich Telefonnummer, E-Mail-Adresse und Raumnummer öffentlich. Wird eine Professorin oder ein Professor neu an eine Hochschule berufen, erfolgt der Datenaustausch zwischen der Personalabteilung und dem Dekanat entweder nur unter Einbezug von Personen, die streng vertrauliche Daten einsehen dürfen, oder zwischen mehreren Personen mit jeweils gleichen Zugriffsrechten. D.h. die Adressdaten werden beispielsweise zwischen Se-

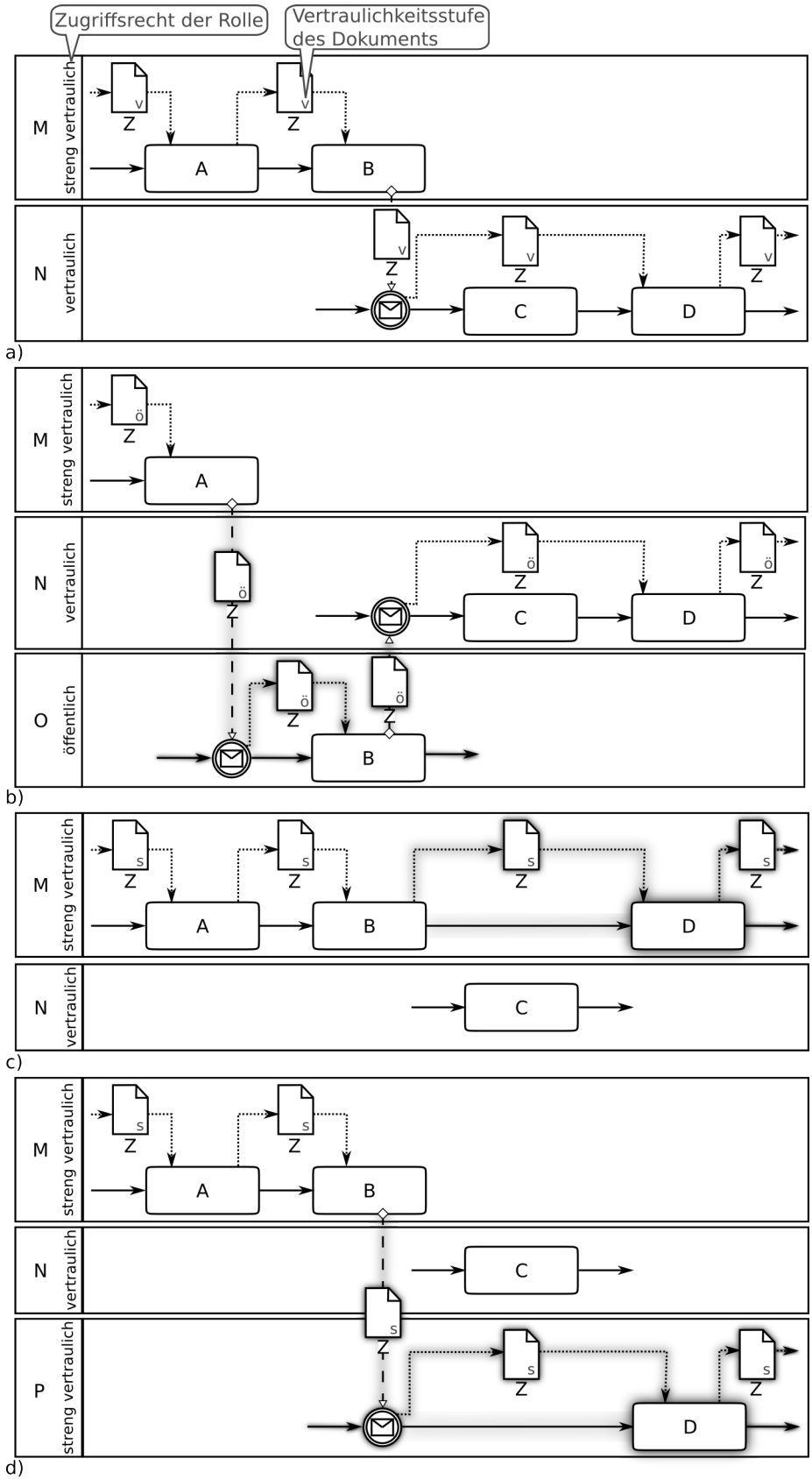


Abbildung 26: Modifizieren der Vertraulichkeit von Datenobjekten eines Prozessmodells

kretariaten, weitere Informationen zwischen Sachbearbeitenden und die streng vertraulichen Daten zwischen dem Dekan und dem Leiter der Personalabteilung ausgetauscht.

4.2.4 Modifizieren der Operationsperspektive

Modifizieren der Abfolge der Datenbearbeitung In einer Prozessvariante kann die Datenverarbeitung modifiziert werden. So kann in einer Sequenz die Reihenfolge der Bearbeitungsschritte verändert werden oder die Art kann modifiziert werden, d.h. beispielsweise kann eine sequenzielle in eine parallele oder alternative Datenverarbeitung umgeformt werden.
Analog zum Ändern der Anordnung von Aktivitäten (siehe Abschnitt Änderung der Anordnung der Aktivitäten) gilt für das Ändern des Datenflusses entsprechend, dass anschließend überprüft werden muss, ob der Kontrollfluss anzupassen ist, und ob die entstandene Kontroll- und Datenflusskombination ausführbar ist.

Übermittlungsart Die Übermittlungsart beschreibt den Austausch der Daten zwischen den am Prozess beteiligten Rollen. Sie bildet die Schnittstelle zwischen den Rollen und beeinflusst, wie die empfangenen Daten weiterverarbeitet werden können.

Beispiel 4.9. (Form, in der die Unterlagen eingereicht werden) In dem in Abbildung 18 gezeigten Prozess werden die Bewerbungen von der Studierendenverwaltung über ein Onlineportal entgegengenommen. Dadurch liegen die Daten zu der Bewerberin bzw. dem Bewerber elektronisch vor und können direkt an die entsprechenden Sachbearbeitenden weitergeleitet werden. In einer Variante des Prozesses könnte vorgesehen werden, dass Bewerbungen auch mittels Papierformularen auf dem Postweg eingereicht werden können. In diesem Fall muss der Prozess um eine Aktivität erweitert werden, in der die Daten manuell in das Datenbanksystem eingeben werden. Der weiter Prozessablauf kann unverändert beibehalten werden.

4.2.5 Modifizieren der Organisationsperspektive

In Varianten können die an dem Prozess beteiligten Rollen Änderungen erfahren, die sich wiederum auf den Datenfluss auswirken können. Werden die ursprünglichen an einem Prozess beteiligten Rollen nicht einfach durch andere Rollen substituiert, sondern ersatzlos gelöscht oder neue Rollen hinzugefügt, wirkt sich dies einerseits auf den Kontrollfluss und andererseits auf den Datenfluss aus. Darüber hinaus kann der Datenfluss davon beeinflusst werden, wenn bestimmte Eigenschaften einer beteiligten Rolle in einer Prozessvariante verändert werden.

Löschen einer Rolle In Prozessen kann es zwei Arten von beteiligten Rollen geben: einfach und mehrfach beteiligte Rollen. Wird eine Mehrfachbeteiligung gelöscht, d.h. die Rolle ist anschließend nur noch einfach beteiligt, wirkt sich dies lediglich auf die Referenzdaten aus, die dadurch nicht mehr benötigt werden. Kontroll- und Datenfluss bleiben jedoch unverändert. Wird eine nur einfach an einem Prozess beteiligte Rolle gelöscht, geht dies damit einher, dass der Kontroll- und Datenfluss dieser Rolle gelöscht wird. Es muss entschieden werden, woher die Informationen, die von dieser Rolle erzeugt wurden, erhalten werden können, oder ob sie für die Variante obsolet sind. Weiter muss festgelegt werden, an wen die Informationen weitergeben werden, die ursprünglich an diese Rolle geschickt wurden, oder ob diese hinfällig sind. Kurz, es muss geprüft werden, wie sich der Datenfluss wieder korrekt herstellen lässt.

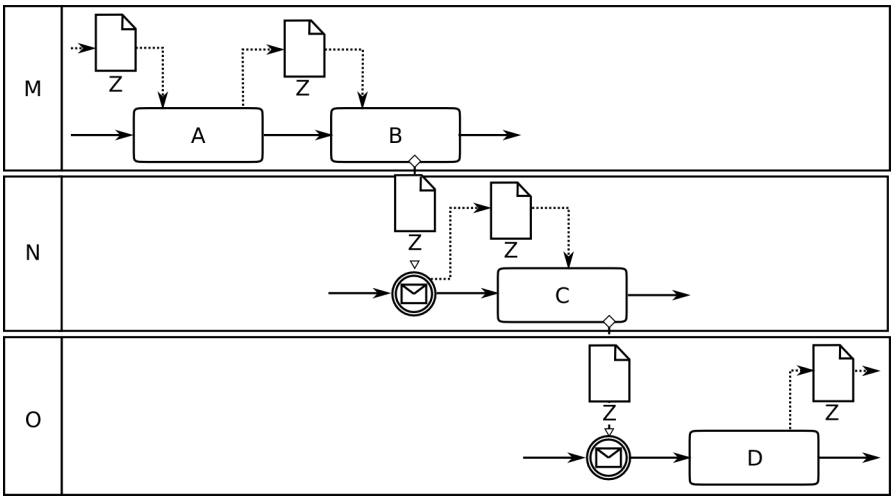


Abbildung 27: Ausgangsprozessmodell für das Löschen einer Rolle

Erläuterung zum Löschen einer Rolle anhand Abbildung 27. Die Aktivitäten A, B, C und D, die Rollen M, N und O sowie das Datenobjekt Z sind Teil eines Geschäftsprozesses. Kontroll- und Datenfluss des Geschäftsprozesses sowie ihr Zusammenspiel sind korrekt. Zur Veranschaulichung wird, ohne Einschränkung der Gültigkeit für andere Anordnungen, ein sequenzieller Kontroll- und Datenfluss zwischen den Aktivitäten angenommen (vgl. Abbildung 27). Das Datenobjekt Z wird zwischen den Rollen ausgetauscht.
Wird die Rolle N gelöscht, kann dies auf den Datenfluss folgende Auswirkungen haben:

- Wird Aktivität C, die von der Rolle ausgeführt wird, ebenfalls gelöscht, sind dieselben Betrachtungen wie beim Löschen einer Aktivität anzustellen.
- Wird Aktivität C nicht gelöscht, ist sie einer bereits bestehenden Rolle oder einer neu am Prozess beteiligten Rolle zuzuweisen.
- Handelt es sich bei N um eine Mehrfachrolle, die zu einer Einfachrolle gelöscht wird, hat dies keine Auswirkungen auf Kontroll- und Datenfluss. Ist die Rolle mit einer mehrfach instanziierten Aktivität verbunden, ist deren Mehrfachinstanziierung einschließlich Referenzdaten zu löschen.

Beispiel 4.10. (Bewerbung auf Zulassung zu einem Studiengang) Die Prozessdarstellung in Abbildung 18 verdeutlicht, dass der Zulassungsprozess für eine Vielzahl von Bewerbenden mit demselben Kontroll- und Datenfluss ausgeführt wird. Dies bedingt auf der Seite der Studierendenverwaltung, dass die Aktivität, die mit einer Bewerberin bzw. einem Bewerber kommuniziert, mehrfach ausgeführt wird. Um die verschiedenen Instanzen der Aktivität unterscheiden zu können, werden Referenzdaten verwendet. Wäre der Prozess nur für eine Bewerberin bzw. einen Bewerber ausgelegt, bliebe der modellierte Kontroll- und Datenfluss bestehen und die Referenzdaten würden entfernt.

Einfügen einer Rolle Es gibt zwei Möglichkeiten, eine Rolle zu einem Prozess hinzuzufügen. Einerseits kann vorgesehen werden, dass eine bereits vorhandene Rolle neu mehrfach beteiligt ist. An-

dererseits kann eine neue Rolle an dem Prozess beteiligt werden, wobei Informationen zwischen den ursprünglich beteiligten Rollen und der hinzugefügten Rolle ausgetauscht werden. Während das Einfügen einer Mehrfachbeteiligung keinen Einfluss auf den modellierten Kontroll- und Datenfluss hat und sich lediglich darin niederschlägt, dass Referenzdaten benötigt werden, wird beim Hinzufügen einer einfach beteiligten Rolle der ursprüngliche Kontrollfluss erweitert und für die neue Rolle ein eigenständiger Kontroll- und Datenfluss modelliert.

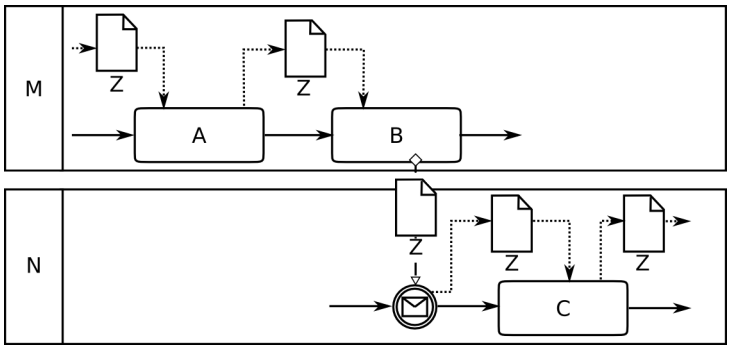


Abbildung 28: Ausgangsprozessmodell für das Einfügen einer Rolle

Erläuterung zum Einfügen einer Rolle anhand Abbildung 28. Die Aktivitäten A, B und C, die Rollen M und N sowie das Datenobjekt Z sind Teil eines Geschäftsprozesses. Kontroll- und Datenfluss des Geschäftsprozesses sowie ihr Zusammenspiel sind korrekt. Zur Veranschaulichung wird, ohne Einschränkung der Gültigkeit für andere Anordnungen, ein sequenzieller Kontroll- und Datenfluss zwischen den Aktivitäten angenommen (vgl. Abbildung 28). Das Datenobjekt Z wird zwischen den Rollen ausgetauscht.
Wird eine Rolle zum Prozess hinzugefügt, kann dies auf den Datenfluss folgende Auswirkungen haben:

- Die Rolle O umfasst eine neue Aktivität D, die in den Datenfluss einbezogen werden muss. Für dieses Szenario gelten dieselben Überlegungen wie beim Einfügen einer Aktivität.
- Eine bereits existierende Aktivität, exemplarisch B, wird der neuen Rolle O zugeordnet, d.h. Rolle M verliert die Zuständigkeit über diese Aufgabe. Der ursprüngliche Datenfluss bleibt bestehen, erstreckt sich jedoch über die neue Rolle.
- Wird eine bestehende Rolle zu einer Mehrfachrolle erweitert, hat dies keine Auswirkungen auf den Kontroll- und Datenfluss. Steht die Rolle mit einer Aktivität einer anderen Rolle in Verbindung, wird diese zu einer mehrfach instanziierten Aktivität geändert und Referenzdaten hinzugefügt.

Beispiel 4.11. (Bewerbung auf Zulassung zu einem Studiengang unter Einbezug der Stiftung für Hochschulzulassung) Bei bundesweit zulassungsbeschränkten Studiengängen müssen sich die Bewerbenden bei der Stiftung für Hochschulzulassung melden. Die Stiftung stellt im Vergleich zum Ausgangsprozess in Abbildung 18 eine neue beteiligte Rolle dar. Der Datenfluss ist in dieser Prozessvariante so gestaltet, dass die Bewerbenden ihre Unterlagen bei der Stiftung einreichen, anstelle sie direkt an die Hochschule zu schicken. Die Stiftung für Hochschulzulassung nimmt die Prüfung

der Unterlagen und den Auswahlprozess vor, d.h. es wird ein neuer Kontroll- und Datenfluss für diese Rolle modelliert, um diese Aufgaben durchzuführen. Absagen werden den Bewerbenden von der Stiftung direkt mitgeteilt. Die Unterlagen der Personen, die zugelassen werden, werden an die entsprechende Hochschule gesandt. Diese nimmt die Daten in ihrem Studierendenverwaltungssystem auf und verschickt die Immatrikulationsunterlagen.

Modifizieren einer Rolle Eine Rolle kann verschiedenen Anpassungen erfahren. Beispielsweise kann die Menge der ihr zugeordneten Akteure verändert werden, oder die Zuständigkeiten einer Rolle, d.h. der zugeordnete Aufgabenbereich, kann angepasst werden. Letzteres wirkt sich nicht nur darauf aus, mit welchen Aktivitäten eine Rolle verbunden werden kann, sondern auch darauf, auf welche Daten die Akteure einer Rolle zugreifen dürfen.

Zugriffsrechte einer Rolle Jede Rolle hat Kompetenzen und damit verbundene Aufgaben in Form von Aktivitäten. Mit den Kompetenzen sind auch Rechte verbunden, auf Daten bis zu einer bestimmten Vertraulichkeitsstufe zugreifen zu dürfen. Bezüglich Zugriff selbst kann wiederum zwischen *nur Lesen*, *nur Schreiben* oder *Lesen und Schreiben* unterschieden werden. Werden die Zugriffsrechte einer Rolle verändert und die Vertraulichkeitseinstufung der Datenobjekte beibehalten, kann dies zur Folge haben, dass weitere Rollen hinzugezogen werden müssen bzw. können, d.h. sich der Kontroll- und Datenfluss des Prozesses ändert.

Erläuterung zu Zugriffsrechte einer Rolle anhand Abbildung 29. Die Aktivitäten A, B, C und D, die Rollen M und N und das Datenobjekt Z sind Teil eines Geschäftsprozesses. Kontroll- und Datenfluss des Geschäftsprozesses sowie ihr Zusammenspiel sind korrekt. Zur Veranschaulichung wird, ohne Einschränkung der Gültigkeit für andere Anordnungen, ein sequenzieller Kontroll- und Datenfluss zwischen den Aktivitäten angenommen. Das Datenobjekt Z wird zwischen den Rollen M und N ausgetauscht. Rollen können über Zugriffsrechte auf öffentliche, vertrauliche oder streng vertrauliche Datenobjekte verfügen. Die Rolle M hat das Zugriffsrecht auf alle Daten, einschließlich streng vertraulicher. Rolle N hat Zugriffsrechte auf vertrauliche Daten. Das Datenobjekt Z ist vertraulich (vgl. Abbildung 29a).

- Erhält die Rolle N Zugriffsrechte auf streng vertrauliche Daten, kann der Prozess entweder
  - wie bisher ablaufen, d.h. Kontroll- und Datenfluss bleiben unverändert oder
  - es wird eine neue Rolle Q in den Prozess einbezogen, die Zugriffsrechte auf vertrauliche Daten besitzt, und an die verschiedene Aktivitäten, die mit dem vertraulichen Datenobjekt Z verbunden sind, delegiert werden (vgl. Abbildung 29b).
- Werden die Zugriffsrechte der Rolle N auf öffentliche Daten eingeschränkt, hat dies zur Folge, dass eine neue Rolle Q mit Zugriffsrecht auf vertrauliche Daten, die Aktivitäten von N übernehmen muss, die mit den vertraulichen Daten Z verbunden sind (vgl. Abbildung 29c).

Beispiel 4.12. (Austausch von Studierendendaten zwischen Studierendenverwaltung und Fakultäten) Studierendendaten sind prinzipiell vertraulich. Ihre Stammdaten dürfen nur von der Studierendenverwaltung geändert werden und Prüfungsdaten nur von der Fakultät, in der die Leistung

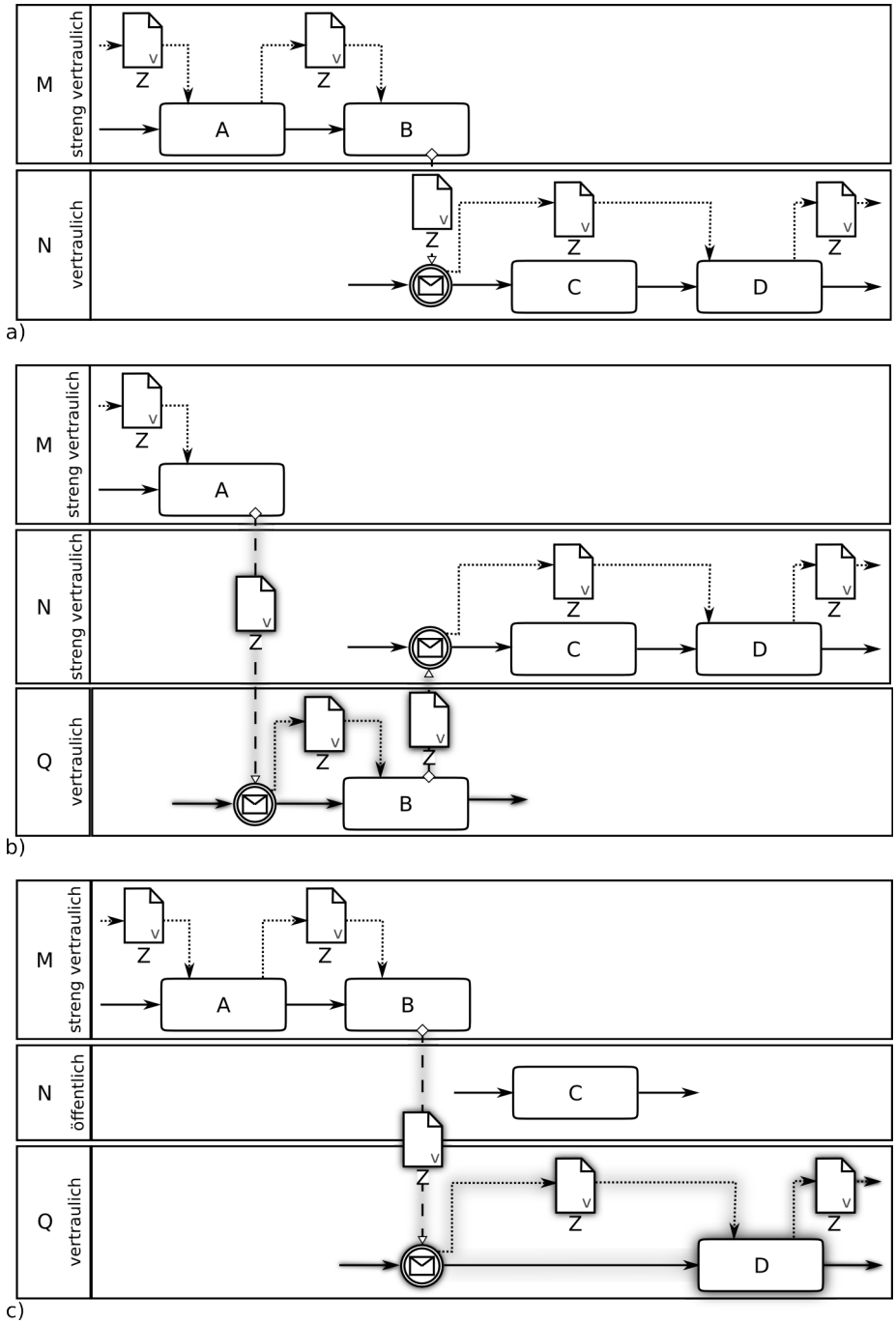


Abbildung 29: Modifizieren der Zugriffsrechte einer Rolle

erworben wurde. Informationen zu Disziplinarverfahren, einem Ausschluss vom Studium oder ähnlichem werden von der Studierendenverwaltung gepflegt und streng vertraulich behandelt. Sachbearbeitende, die in den Fakultäten mit Studierendendaten arbeiten, erhalten prinzipiell lesende Zugriffsrechte auf vertrauliche Daten. Die Studiendekaninnen und Studiendekane der Fakultäten dürfen streng vertrauliche Informationen einsehen. Abhängig von der fakultätsinternen Organisation ist denkbar, dass streng vertrauliche Daten an die Sachbearbeitenden weitergeben werden sollen. In einer solchen Prozessvariante sind die Zugriffsrechte der Sachbearbeitenden und der Kontroll- und Datenfluss entsprechend anzupassen.

5 Fachliche Anforderungen und Herausforderungen

Aus den vorangegangenen Überlegungen zu Datenflussvarianten lassen sich verschiedene fachliche Anforderungen und Herausforderungen ableiten. An erster Stelle ist die Korrektheit der Prozessvarianten zu nennen, d.h. einerseits darf der Datenfluss keine Anomalien aufweisen und andererseits müssen Kontroll- und Datenfluss zueinander in Einklang stehen, so dass beispielsweise Verklemmungen ausgeschlossen werden können. Eine weitere wichtige Anforderung an Prozessvarianten ist, dass sie wartbar und überschaubar bleiben. Wird während der Evolutionsphase eines Prozesses festgestellt, dass er den Anforderungen nicht mehr genügt, da sich beispielsweise die juristischen Rahmenbedingungen geändert haben, soll eine Anpassung aller betroffenen Prozessvarianten einfach erfolgen können.

5.1 Korrektheit

Die Bedeutung der Datenflusskorrektheit wurde schon früh erkannt. Im Weiteren sollen verschiedene Arbeiten zur Korrektheit bzw. zu Datenflussanomalien vorgestellt werden.
In [SOSF04], einer der ersten Forschungsarbeiten zu diesem Thema, wurde untersucht, welche Datenflussinkonsistenzen auftreten können. Es wurden folgende Anomalien identifiziert:

- *Unnötige Daten (Redundat Data).* Diese Anomalie liegt vor, wenn eine Aktivität Daten generiert, die weder von einer nachfolgenden Aktivität benötigt werden, noch an eine andere Rolle oder externe Stelle weitergegeben werden, noch Ausgabe des Prozesses sind.

- *Verlorene Daten (Lost Data).* Wird der Wert eines Datenobjekts überschrieben, ohne dass der aktuelle Wert zuvor von einer Aktivität gelesen wurde, handelt es sich um verlorene Daten. Dieses Szenario kann beispielsweise beim parallelen Schreiben eines Datenobjekts eintreten.

- *Fehlende Daten (Missing Data).* Fehlende Daten beschreiben den Umstand, dass eine Aktivität ein Datenobjekt als Eingabe benötigt, sie jedoch nicht mit diesem versorgt wird.

- *Fehlangepasste Daten (Mismatched Data).* Fehlangepasste Daten liegen vor, wenn mehrere Aktivitäten dasselbe Datenobjekt verarbeiten, sie es aber in unterschiedlichen Datenstrukturen erwarten, wobei eine entsprechende Transformation der Daten nicht im Prozess vorgesehen ist.

- *Inkonsistente Daten (Inconsistent Data).* Inkonsistente Daten treten beispielsweise dann auf, wenn ein Datenobjekt durch Auslesen einer Information aus einer Datenbank, die einer Vielzahl von Prozessen zugänglich ist, initialisiert wird. Werden nach der Initialisierung des Datenobjekts die Daten der Datenbank von einem anderen Prozess geändert, ohne dass dies an den laufenden Prozess weitergegeben wird, arbeitet der Prozess mit veralteten Informationen.

- *Fehlgeleitete Daten (Misdirected Data).* Widersprechen sich Kontroll- und Datenfluss eines Prozesses, wird von fehlgeleiteten Daten gesprochen.

- *Unzureichende Daten (Insufficient Data).* Es liegen unzureichende Daten vor, wenn die Informationen, mit denen eine Aktivität über ihre Datenobjekte versorgt wird, nicht genügen, um die Aktivität auszuführen.

In [SZNS06] werden drei grundlegende Kategorien von Anomalien vorgestellt: fehlende Daten *(Missing Data)*, unnötige Daten *(Redundant Data)* und widersprüchliche Daten *(Conflicting Data)*. Es wird festgehalten, dass die Anomalien von [SOSF04] mit Ausnahme der inkonsistenten Daten durch diese drei Kategorien repräsentiert werden können. Nach Auffassung von [SZNS06] stellen inkonsistente Daten keine konzeptionelle Datenflussanomalien dar. Zusätzlich zur konzeptionellen Betrachtung wird einen Mechanismus zur Korrektheitsprüfung entwickelt. Dieser basiert auf einer Datenflussmatrix.

Auch die Überlegungen zu Datenfluss-Antipattern [TAS09] setzen ebenfalls auf den von [SOSF04] vorgestellten Anomalien auf. Vier der ursprünglichen Kategorien werden übernommen und teilweise verfeinert sowie drei Kategorien hinzugefügt, die das Löschen bzw. Zerstören von Datenobjekten thematisieren. Dies resultiert in folgenden Kategorien: fehlende Daten *(Missing Data)*, stark unnötige Daten *(Strongly Redundant Data)*, schwach unnötige Daten *(Weakly Redundant Data)*, stark fehlende Daten *(Strongly Lost Data)*, schwach fehlende Daten *(Weakly Lost Data)*, inkonsistente Daten *(Inconsistent Data)*, niemals zerstörte Daten *(Never Destroyed)*, zweimal zerstörte Daten *(Twice Destroyed)* und nicht rechtzeitig gelöschte Daten *(Not Deleted On Time)*. Die „schwachen“ Anomalien zeichnen sich dadurch aus, dass sie nur in gewissen Ausführungsszenarien auftreten, während „starke“ Anomalien in jedem Ausführungsszenario vorliegen.

Unabhängig von den eingangs vorgestellten Anomalien stellt [ADL10] drei Kategorien von Datenflussfehlern und deren Lösungen vor: zu einschränkende Vorbedingungen *(Too Restrictive Preconditions)*, implizites Routing *(Implicit Routing)* und implizite Einschränkung der Ausführungsreihenfolge *(Implicit Constraints On The Execution Order)*. Ersteres kann auf die Kategorie der fehlenden Daten und die beiden anderen auf die Kategorie der fehlgeleiteten Daten abgebildet werden.

Das Zusammenspiel bzw. die Korrektheit von Kontroll- und Datenfluss, d.h. das Verhindern von fehlgeleiteten Daten ist auch Gegenstand von anderen aktuellen Betrachtungen. Beispielsweise stellt [TAS08] einen auf Workflow-Netzen basierenden Ansatz vor, um Kontroll- und Datenfluss gleichzeitig zu analysieren. Ein für die Betrachtung von Prozessvarianten sehr interessanter Ansatz wird in [RM09] vorgestellt. Er beschreibt, wie die Datenflusskorrektheit in adaptiven Workflow Systemen in der Modellierungs- und Ausführungsphase sichergestellt werden kann.

Zusammenfassend lassen sich für Prozessvarianten folgende Korrektheitsanforderungen formulieren:

- Der Kontrollfluss muss korrekt sein.

- Es dürfen keine Datenflussanomalien vorliegen. Dies beinhaltet insbesondere durch das Ausschließen von fehlgeleiteten Daten, dass das Zusammenspiel von Kontroll- und Datenfluss nicht zu Fehlern führen darf. Es lassen sich folgende konkreten Forderungen formulieren:

  - Die Kontroll- und die Datenflussrichtung von operativen Daten müssen übereinstimmen.

  - Mehrere Aktivitäten dürfen nicht parallel dasselbe Datenobjekt schreiben.
  - Bei alternativem Kontrollfluss darf der Datenfluss nicht zwischen den Kontrollflusspfaden wechseln.
  - Ein alternativer Datenfluss von Datenobjekten beginnt immer auf einem alternativen Kontrollflusspfad.
  - Für die Kontrollflussentscheidung an `OR`- und `XOR`-Verzweigungsknoten müssen die Datenobjekte berücksichtigt werden, die auf mindestens einem der von ihnen ausgehenden Kontrollflusspfade verwendet werden. Eine Ausnahme sind die Datenobjekte, die auf einem solchen Pfad neu erstellt werden.

- Die Korrektheit sollte einfach und mit geringem, bestenfalls minimalen, Aufwand prüfbar sein. Dabei kann einer der folgenden Mechanismen verwendet werden:
  - Der ursprüngliche Prozess und jede aus ihm entstehende Prozessvariante werden vollständig auf korrekten Kontroll- und Datenfluss geprüft.
  - Der ursprüngliche Prozess wird auf korrekten Kontroll- und Datenfluss geprüft. Bei den Prozessvarianten werden lediglich die neu modellierten Prozessteile und ihre Abhängigkeiten zum ursprünglichen Prozessmodell auf Korrektheit geprüft.
  - Zur Modellierung von Prozessvarianten wird eine Menge von zulässigen Änderungen definiert, die stets zu einem korrekten Prozessmodell führen.

5.2 Überschaubare Komplexität und Wartbarkeit

Varianten entstehen dadurch, dass aufgrund spezifischer Anforderungen gezielt von dem ursprünglich vorgegebenen Prozessmodell abgewichen wird. Beispielsweise ist die Zulassung zu einem Studiengang in gewissen Prozessabschnitten für jede Studienstufe (Bachelor, Master, Doktorat) gesondert gestaltet. Dabei können innerhalb dieser Studienstufen weitere Prozessvarianten existieren. Zum Beispiel werden die Bewerbenden nach der Herkunft ihres Hochschulzulassungsausweises (Deutschland, EU, übriges Ausland) unterschieden. Um die Komplexität bei der Modellierung, aber auch bei der Evolution in Grenzen zu halten, sollten Prozessvarianten nicht vollständig voneinander unabhängig sein, sondern auf einer gemeinsamen Basis aufbauen und die Unterschiede zueinander spezifizieren. Vorteil eines solchen Vorgehens ist, dass sich eine Evolution der gemeinsamen Basis auf alle Prozessvarianten auswirkt und nicht jedes einzelne Variantenprozessmodell separat angepasst werden muss. Die Wartbarkeit der Prozessvarianten steht im direkten Zusammenhang mit der Komplexität des Prozessmodells.

Tritt der Lebenszyklus der Prozessvarianten in die Phase der Analyse & Evolution liegen neben den Prozessvariantenmodellen, der ursprüngliche Basisprozess und die darauf definierten Modifikationen zur Variantenkonfiguration vor. Einerseits liegt die Herausforderung darin, die Anpassungen am Basisprozess bzw. den Modifikationen zur Variantenkonfiguration geeignet auf die Prozessvariantenmodelle zu übertragen, andererseits können die im Zuge einer Evolution vorgenommenen Änderungen unterschiedlich auf die laufenden Prozessinstanzen propagiert werden [JH98, AJ00, Aal03, QW07, RRD03, RRD04]:

- *Neustart der Prozessinstanz mit dem neuen Prozessmodell (Restart, Backward Recovery).* Die laufenden Prozessinstanzen werden gestoppt und ihre Auswirkungen rückgängig gemacht. Anschließend werden sie mit dem neuen Prozessmodell gestartet (vgl. Abbildung 30a).

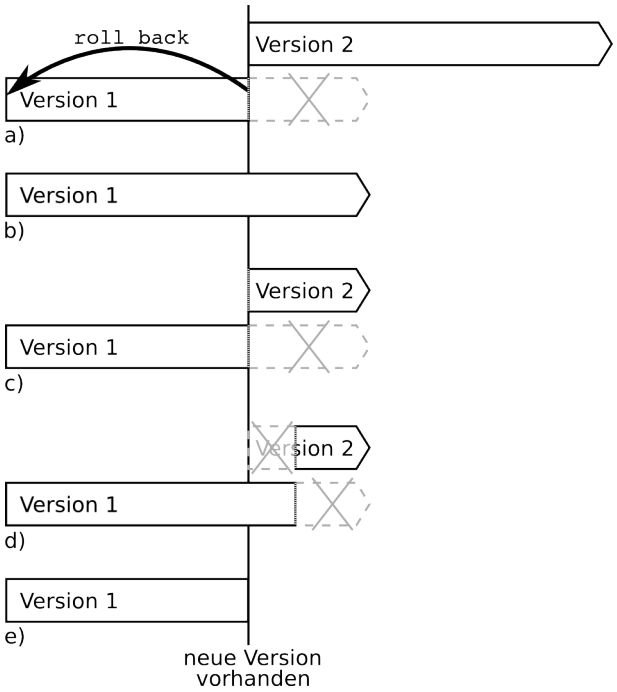


Abbildung 30: Verbreitungsalternativen neuer Prozessversionen

- *Unverändertes Ausführen der Prozessinstanz mit dem bisherigen Prozessmodell (Proceed, Lazy Propagation).* Alle bereits laufenden Prozessinstanzen werden ohne Berücksichtigung des neuen Prozessmodells zu Ende geführt (vgl. Abbildung 30b). Werden neue Instanzen gestartet, verwenden diese das neue Prozessmodell. Bei dieser Art der Ausbreitung eines neuen Prozessmodells existieren verschiedene Varianten eines Prozesses zur selben Zeit.

- *Transfer der Prozessinstanz auf das neue Prozessmodell (Transfer, Eager Propagation).* Die laufenden Prozessinstanzen werden auf das neue Prozessmodell transferiert (vgl. Abbildung 30c). Dieser Ansatz kann verfeinert werden, indem nur der *Transfer einer Auswahl von Prozessinstanzen (Selective Transfer)* vorgenommen wird. Beispielsweise können nur die Prozessinstanzen auf das neue Prozessmodell transferiert werden, die sich in einem bestimmten Ausführungszustand befinden (vgl. Abbildung 30d). Dies hat zur Folge, dass der Transfer zu unterschiedlichen Zeitpunkten erfolgen kann.

- *Abbruch der Prozessinstanz im Workflow Managementsystem (Forward Recovery).* Sobald ein neues Prozessmodell zur Verfügung steht, werden die laufenden Prozessinstanzen gestoppt und außerhalb des Workflow Managementsystems abgewickelt (vgl. Abbildung 30e).

Der Vollständigkeit halber sollen erwähnt werden, dass darüber hinaus in [JH98] die *Anpassung einer Instanz und etwaige Ausbreitung auf alle Instanzen (Local Modification And Upward Propagation)* und in [AJ00] das *Pausieren einer Prozessinstanz (Detour)* angeführt werden. Diese Szenarien beschreiben, wie mit Ad-hoc-Änderungen bzw. unerwarteten Ereignissen eines Prozesses umgegangen werden kann. Zudem wird in [JH98] mit dem *Mischen der Prozessmodelle (Merging)* ein weiterer Ansatz vorgeschlagen, wie ein neues Prozessmodell auf mehrere bestehende Prozessmodellvarianten ausgebreitet werden kann. Die so entstanden neuen Variantenprozessmodelle können nach einer der

beschriebenen Methoden auf die laufenden Instanzen ausgebreitet werden.

Für eine überschaubare Komplexität und Wartbarkeit des ursprünglichen Prozesses und seiner Varianten wird folgendes gefordert:

- Die Varianten müssen einen gemeinsamen Basisprozess besitzen und über vorgegebene Regeln aus diesem modellierbar sein.
- Es muss zu jeder Zeit feststellbar sein, wie viele Prozessvarianten existieren und wie sie aus dem Basisprozess abgeleitet wurden.
- Der Basisprozess und die darauf definierten Modifikationen sollen keine Redundanz aufweisen. Dies bedeutet insbesondere, dass jede Aktivität genau einmal im Basisprozess vorkommen kann und die Modifikationen, die auf einem Basisprozess vorgenommen werden, miteinander kombinierbar sind.

6 Diskussion

Die vorangegangenen Betrachtungen haben gezeigt, dass Datenflussvarianten durch Modifikationen in den fünf Workflow Perspektiven entstehen können. In Tabelle 5 ist dies zusammenfassend dargestellt, indem für jede vorgestellte Änderung festgehalten wird, welche Prozessvariantenart sie zwingend bzw. optional auslöst.

Durch diese Zusammenfassung wird zudem deutlich, dass sich eine Modifikation in der Regel nicht nur auf eine Variantenart auswirkt, sondern mehrere gleichzeitig betreffen kann. Wird beispielsweise eine Aktivität eingefügt, die mit einem Datenobjekt verbunden ist, so beeinflusst dies neben dem Kontrollfluss auch den Datenfluss, d.h. diese Modifikation schlägt sich sowohl in einer Kontrollflussvariante als auch einer Datenflussvariante nieder. In der Tabelle ist beim Einfügen einer Aktivität jedoch keine zwingende Auswirkung auf den Datenfluss vermerkt, da die einzufügende Aktivität nicht unbedingt mit Daten verbunden sein muss. In einem solchen Fall, besteht keine Notwendigkeit den ursprünglichen Datenfluss zu modifizieren.

7 Zusammenfassung und Ausblick

Um in die Thematik der Datenflussvarianten einzuführen, wurden als Grundlagen der Prozesslebenszyklus, die möglichen Prozessanpassungen und die Daten in Prozessen vorgestellt. Des Weiteren wurde beschrieben, welche Ansätze zum Variantenmanagement bereits existieren. Dabei wurde zwischen der Variantenmodellierung und dem Erkennen von Varianten, d.h. dem Feststellen der strukturellen und semantischen Ähnlichkeit zwischen Prozessen, unterschieden. Die Variantenmodellierung selbst wurde in zwei Ansätze unterteilt: Transformation des Basisprozesses und Selektion aus dem Basisprozess. Bei der Transformation werden Prozessfragmente zum Basisprozess hinzugefügt. Je nach Modellierungsansatz können jedoch auch vorhandene Prozessbestandteile verschoben oder gelöscht werden. Bei Ansatz der Selektion umfasst der Basisprozess alle gewünschten Prozessvarianten, d.h. er stellt ihre Obermenge dar. Die Selektion besteht darin, Teile des Basisprozesses auszulassen bzw. zu löschen.
Der vorgestellte Ausschnitt aus den aktuellen Forschungsarbeiten auf dem Gebiet des Variantenmanagements zeigte, dass der Fokus sehr stark auf dem Kontrollfluss liegt. Es existieren jedoch bereits vereinzelt Ansätze, darauf aufbauend den Datenfluss in die Betrachtungen einzubeziehen.

	Kontrollfluss-variante	Datenfluss-variante	organisatorische Variante
Funktionsperspektive			
Löschen einer Aktivität	√	möglich	möglich
Einfügen einer Aktivität	√	möglich	möglich
Modifizieren einer Aktivität	möglich	möglich	möglich
Verhaltensperspektive			
Modifizieren der Reihenfolge der Aktivitäten	√	möglich	
Datenbasiertes Routing	√	√	
Informationsperspektive			
Löschen eines Datenobjekts	möglich	√	
Einfügen eines Datenobjekts	möglich	√	
Modifizieren eines Datenobjekts:			
• Visibilität		√	
• Datenstruktur/Granularität		√	
• Vertraulichkeit	möglich	√	möglich
Operationsperspektive			
Modifizieren der Reihenfolge der Datenbearbeitung	möglich	√	
Übermittlungsart	möglich	möglich	möglich
Organisationsperspektive			
Löschen einer Rolle	möglich	möglich	√
Einfügen einer Rolle	möglich	möglich	√
Modifizieren einer Rolle:			
• Kompetenzen/Rechte	möglich	möglich	√

Tabelle 5: Einfluss von Änderungen in den Workflow Perspektiven auf die zwingend bzw. optional entstehenden Variantenarten

Im Weiteren wurde untersucht wie Datenflussvarianten entstehen können. Dazu wurde analysiert, wie sich Modifikationen der fünf Workflow Perspektiven auf den Datenfluss auswirken können. Diese Betrachtung zeigten, dass Prozessvarianten meist mehrere Dimensionen umfassen, d.h. eine Kombination von Kontrollflussvariante, Datenflussvariante und organisatorischer Variante vorliegt. Schließlich wurde dargelegt, welche Anforderungen und Herausforderungen die Modellierung von Datenflussvarianten mit sich bringen. Einerseits muss Korrektheit des Datenflusses sichergestellt werden und andererseits sollte der Modellierungsansatz eine überschaubare Komplexität aufweisen und dadurch wartbar bleiben, d.h. den Prozesslebenszyklus und damit insbesondere die Evolution der Prozessvarianten geeignet unterstützen.

Im nächsten Schritt soll untersucht werden wie Datenflussvarianten modelliert werden können, um den geschilderten Anforderungen und Herausforderungen zu genügen. Das Augenmerk soll dabei vorerst auf der Korrektheit liegen.

Literatur

[Aal03] Wil M. P. van der Aalst. Inheritance of business processes: A journey visiting four notorious problems, 2003.

[AB02] Wil M. P. van der Aalst and Twan Basten. Inheritance of workflows – an approach to tackling problems related to change. *Theoretical Computer Science*, 270:125–203, 2002.

[ADG+06] Wil M. P. van der Aalst, Alexander Dreiling, Florian Gottschalk, Michael Rosemann, and Monique H. Jansen-Vullers. Configurable process models as basis for reference modeling. In Christoph Bussler and Armin Haller, editors, *Business Process Management Workshops, BPM 2005 International Workshops, BPI, BPD, ENEI, BPRM, WSCOBPM, BPS, Nancy, France, September 5, 2005, Revised Selected Papers*, volume 3812 of *Lecture Notes in Computer Science*, pages 512–518. Springer, 2006.

[ADG+08] Wil M. P. van der Aalst, Marlon Dumas, Florian Gottschalk, Athur H.M. ter Hofstede, Marcello La Rosa, and Jan Mendling. Correctness-preserving configuration of business process models. In J. Fiadeiro and P. Inverardi, editors, *Fundamental Approaches to Software Engineering (FASE 2008), Budapest, Hungary*, volume 4961 of *Lecture Notes in Computer Science*, pages 46–61. Springer, 2008.

[ADG+10] Wil M. P. van der Aalst, Marlon Dumas, Florian Gottschalk, Athur H.M. ter Hofstede, Marcello La Rosa, and Jan Mendling. Preserving correctness during business process model configuration. *Formal Aspects of Computing (FACS)*, 22:459–482, May 2010.

[ADL10] Ahmed Awad, Gero Decker, and Niels Lohmann. Diagnosing and repairing data anomalies in process models. In Stefanie Rinderle-Ma, Shazia Sadiq, and Frank Leymann, editors, *Business Process Management Workshop, BPM 2009 International Workshop, Ulm, Germany, September 2009, Revised Papers*, volume 43 of *Lecture Notes in Business Information Processing*, pages 5–16. Springer, March 2010.

[AJ00] Wil M. P. van der Aalst and Stefan Jablonski. Dealing with workflow change: Identification of issues and solutions. *International Journal of Computer Systems Science and Engineering*, 15(5):267–276, September 2000.

[APW08] Ahmed Awad, Artem Polyvyanyy, and Mathias Weske. Semantic querying of business process models. In *12th International IEEE Enterprise Distributed Object Computing Conference, ECOC 2008, 15–19 September 2008, Munich, Germany*, pages 85–94, 2008.

[AWW03] Wil M. P. van der Aalst, Mathias Weske, and Guido Wirtz. Advanced topics in workflow management: Issues, requirements, and solutions. *Journal of Integrated Design and Process Science*, 7(3):49–77, 2003.

[CKO92] Bill Curtis, Marc I. Kellner, and Jim Over. Process modeling. *Communications of the ACM*, 35(9):75–90, 1992.

[DDGB09] Remco M. Dijkman, Marlon Dumas, and Luciano García-Bañuelos. Graph matching algorithms for business process model similarity search. In Umeshwar Dayal, Johann Eder, Jana Koehler, and Hajo A. Reijers, editors, *Business Process Management, 7th International Conference, BPM 2009, Ulm, Germany, September 8–10, 2009. Proceedings*, volume 5701 of *Lecture Notes in Computer Science*, pages 48–63. Springer, 2009.

[DDM08] Boudewijn F. van Dongen, Remco M. Dijkman, and Jan Mendling. Measuring similarity between business process models. In Zohra Bellahsene and Michel Léonard, editors, *Advanced Information Systems Engineering, 20th International Conference, CAiSE 2008, Montpellier, France, June 16–20, 2008, Proceedings*, volume 5074 of *Lecture Notes in Computer Science*, pages 450–464. Springer, 2008.

[Eur07] Auf dem Wege zum Europäischen Hochschulraum: Antworten auf die Herausforderungen der Globalisierung, London, 2007. Available from: http://www.bmbf.de/pub/Londoner_Kommunique_Bologna_d.pdf [cited March 9, 2010].

[Gab10] Gabler Wirtschaftslexikon, Stichwort Geschäftsprozess, 2010. Available from: http://wirtschaftslexikon.gabler.de/Archiv/5598/geschaeftsprozess-v8.html [cited November 20, 2010].

[GAJVL08] Florian Gottschalk, Wil M. P. van der Aalst, Monique H. Jansen-Vullers, and Marcello La Rosa. Configurable workflow models. *International Journal of Cooperative Information Systems (IJCIS)*, 17:177–221, 2008.

[GRMR+08] Christian W. Günther, Stefanie Rinderle-Ma, Manfred Reichert, Wil M. P. van der Aalst, and Jan Recker. Using process mining to learn from process changes in evolutionary systems. *International Journal of Business Process Integration and Management, Special Issue on Business Process Flexibility*, 3(1):61–78, 2008.

[GWJV+09] Florian Gottschalk, Teun A. C. Wagemakers, Monique H. Jansen-Vullers, Wil M. P. van der Aalst, and Marcello La Rosa. Configurable process models: Experiences from a municipality case study. In Pascal van Eck, Jaap Gordijn, and Roel Wieringa, editors, *Advanced Information Systems Engineering, 21st International Conference, CAiSE 2009, Amsterdam, The Netherlands, June 8–12, 2009. Proceedings*, volume 5565 of *Lecture Notes in Computer Science*, pages 486–500. Springer, 2009.

[Hal09] Alena Hallerbach. *Management von Prozessvarianten*. PhD thesis, Universität Ulm, 2009.

[HBR08a] Alena Hallerbach, Thomas Bauer, and Manfred Reichert. Context-based configuration of process variants. In *3rd International Workshop on Technologies for Context-Aware Business Process Management, TCoB 2008, Barcelona, Spain, June 12, 2008. Proceedings*, pages 31–40, 2008.

[HBR08b] Alena Hallerbach, Thomas Bauer, and Manfred Reichert. Managing process variants in the process life cycle. In José Cordeiro and Joaquim Filipe, editors, *ICEIS 2008 – Proceedings of the 10th International Conference on Enterprise Information Systems, Volume ISAS-2, Barcelona, Spain, June 12–16, 2008*, pages 154–161, 2008.

[HBR08c] Alena Hallerbach, Thomas Bauer, and Manfred Reichert. Modellierung und Darstellung von Prozessvarianten in Provop. In *Modellierung'08*, pages 41–56, 2008.

[HBR09] Alena Hallerbach, Thomas Bauer, and Manfred Reichert. Guaranteeing soundness of configurable process variants in Provop. In Birgit Hofreiter and Hannes Werthner, editors, *11th IEEE Conference on Commerce and Enterprise Computing, CEC 2009, Vienna, Austria, July 20-23, 2009*, pages 98–105. IEEE Computer Society, 2009.

[HBR10a] Alena Hallerbach, Thomas Bauer, and Manfred Reichert. Capturing variability in business process models: the Provop approach. *Journal of Software Maintenance*, 22(6–7):519–546, 2010.

[HBR10b] Alena Hallerbach, Thomas Bauer, and Manfred Reichert. Configuration and management of process variants. In Jan Brocke and Michael Rosemann, editors, *Handbook on Business Process Management I*, International Handbooks Information System, chapter 11, pages 237–255. Springer, 2010.

[Hol95] David Hollingsworth. *Workflow Management Coalition – The Workflow Reference Model*. Workflow Management Coalition, January 1995. Document Number TC00-1003 – Document Status - Issue 1.1.

[JH98] Gregor Joeris and Otthein Herzog. Managing evolving workflow specifications. In *Proceedings of the 3rd IFCIS International Conference on Cooperative Information Systems, New York City, New York, USA, August 20–22, 1998, Sponsored by IFCIS, The International Foundation on Cooperative Information Systems*, pages 310–321. IEEE Computer Society, 1998.

[LADH09] Marcello La Rosa, Wil M. P. van der Aalst, Marlon Dumas, and Athur H.M. ter Hofstede. Questionaire-based variability modeling for system configuration. *Software and System Modeling*, 8(2):251–274, 2009.

[LDH+08] Marcello La Rosa, Marlon Dumas, Athur H.M. ter Hofstede, Jan Mendling, and Florian Gottschalk. Beyond control-flow: Extending business process configuration to roles and objects. In Qing Li, Stefano Spaccapietra, and Eric Yu, editors, *Proceedings 27th International Conference on Conceptual Modeling (ER 2008), Barcelona, Spanien*, pages 199–215. Springer, 2008.

[LDKD10] Marcello La Rosa, Marlon Dumas, Reina Käärik, and Remco M. Dijkman. Merging business process models. In *Proceedings of the 18th International Conference on Cooperative Information Systems (CoopIS), Kreta, Griechenland*, volume 6426 of *Lecture Notes in Computer Science*, pages 96–113. Springer, 2010.

[LDUD09] Marcello La Rosa, Marlon Dumas, Reina Uba, and Remco M. Dijkman. Business process model merging: An approach to business process consolidation, 2009.

[LM07] Richard Lenz and Reichert Manfred. IT support for healthcare processes – premises, challenges, perspectives. *Data & Knowledge Engineering*, 61(1):39–58, 2007.

[LRDHM11] Marcello La Rosa, Marlon Dumas, Arthur H. M. ter Hofstede, and Jan Mendling. Configurable multi-perspective business process models. *Information Systems*, 36(2):313–340, 2011.

[LRW08a] Chen Li, Manfred Reichert, and Andreas Wombacher. Discovering reference process models by mining process variants. In *2008 IEEE International Conference on Web Services (ICWS 2008), September 23–26, 2008, Beijing, China*, pages 45–53. IEEE Computer Society, 2008.

[LRW08b] Chen Li, Manfred Reichert, and Andreas Wombacher. On measuring process model similarity based on high-level change operations. In Qing Li, Stefano Spaccapietra, Eric S. K. Yu, and Antoni Olivé, editors, *Conceptual Modeling - ER 2008, 27th International Conference on Conceptual Modeling, Barcelona, Spain, October 20-24, 2008. Proceedings*, volume 5231 of *Lecture Notes in Computer Science*, pages 248–264. Springer, 2008.

[LRW09a] Chen Li, Manfred Reichert, and Andreas Wombacher. Discovering reference models by mining process variants using a heuristic approach. In Umeshwar Dayal, Johann Eder, Jana Koehler, and Hajo A. Reijers, editors, *Business Process Management, 7th International Conference, BPM 2009, Ulm, Germany, September 8-10, 2009. Proceedings*, volume 5701 of *Lecture Notes in Computer Science*, pages 344–362. Springer, 2009.

[LRW09b] Chen Li, Manfred Reichert, and Andreas Wombacher. What are the problem makers: Ranking activities according to their relevance for process changes. In *IEEE 7th International Conference on Web Services, ICWS 2009, Los Angeles, CA, USA, 6-10 July 2009*, pages 51–58. IEEE Computer Society, 2009.

[LRW10] Chen Li, Manfred Reichert, and Andreas Wombacher. The MinAdept clustering approach for discovering reference process models out of process variants. *International Journal of Cooperative Information Systems*, 19(3–4):159–203, 2010.

[LS06] Ruopeng Lu and Shazia Wasim Sadiq. On managing process variants as an information resource. Technical Report No.464, School of Information Technology and Electrical Engineering, University of Queensland, 2006.

[LSG09] Ruopeng Lu, Shazia Wasim Sadiq, and Guido Governatori. On managing business processes variants. *Data & Knowledge Engineering*, 68(7):642–664, 2009.

[MHHR06] Dominic Müller, Joachim Herbst, Markus Hammori, and Manfred Reichert. IT support for release management processes in the automotive industry. In Schahram Dustdar, José Luiz Fiadeiro, and Amit P. Sheth, editors, *Business Process Management, 4th International Conference, BPM 2006, Vienna, Austria, September 5–7, 2006, Proceedings*, volume 4102 of *Lecture Notes in Computer Science*, pages 368–377. Springer, 2006.

[MRB08] Bela Mutschler, Manfred Reichert, and Johannes Bumiller. Unleashing the effectiveness of process-oriented information systems: Problem analysis, critical success factors, and implications. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 38(3):280–291, 2008.

[OF06] Margit Osterloh and Jetta Frost. *Prozessmanagement als Kernkompetenz*. Gabler, Wiesbaden, 5th edition, 2006.

[OMG11] OMG (Object Management Group). *Business Process Model and Notation (BPMN) Version 2.0*. OMG, January 2011. formal/2011-01-03. Available from: http://www.omg.org/spec/BPMN/2.0/ [cited March 19, 2011].

[PDKL09] Kerstin Pfitzner, Gero Decker, Oliver Kopp, and Frank Leymann. Web service choreography configurations for BPMN. In Elisabetta Nitto and Matei Ripeanu, editors, *Service-Oriented Computing - ICSOC 2007 Workshops*, pages 401–412. Springer, 2009.

[QW07] Z. M. Qiu and Y. S. Wong. Dynamic workflow change in pdm systems. *Computers in Industry*, 58:453–463, June 2007.

[RD97] Manfred Reichert and Peter Dadam. A framework for dynamic changes in workflow management systems. In *8th International Conference and Workshop on Database and Expert Systems Applications, DEXA 1997, Toulouse, France, September 1–5, 1997, Proceedings*, pages 42–48, 1997.

[Rei00] Manfred Reichert. *Dynamische Ablaufänderungen in Workflow-Management-Systemen*. PhD thesis, Universität Ulm, 2000.

[RHEA04] Nick Russell, Arthur H.M. ter Hofstede, David Edmond, and Wil M. P. van der Aalst. Workflow data patterns. Technical Report FIT-TR-2004-01, Queensland University of Technology, 2004.

[RM09] Stefanie Rinderle-Ma. Data flow correctness in adaptive workflow systems. *EMSIA Forum*, 29(2):25–35, 2009.

[RMRW08] Stefanie Rinderle-Ma, Manfred Reichert, and Barbara Weber. On the formal semantics of change patterns in process-aware information systems. In Qing Li, Stefano Spaccapietra, Eric S. K. Yu, and Antoni Olivé, editors, *Conceptual Modeling – ER 2008, 27th International Conference on Conceptual Modeling, Barcelona, Spain, October 20-24, 2008. Proceedings*, volume 5231 of *Lecture Notes in Computer Science*, pages 279–293. Springer, 2008.

[RR06] Stefanie Rinderle and Manfred Reichert. Data-driven process control and exception handling in process management systems. In Eric Dubois and Klaus Pohl, editors, *Advanced Information Systems Engineering, 18th International Conference, CAiSE 2006, Luxembourg, Luxembourg, June 5–9, 2006, Proceedings*, volume 4001 of *Lecture Notes in Computer Science*, pages 273–287. Springer, 2006.

[RRD03] Manfred Reichert, Stefanie Rinderle, and Peter Dadam. On the common support of workflow type and instance changes under correctness constraints. In Robert Meersman, Zahir Tari, and Douglas C. Schmidt, editors, *On The Move to Meaningful Internet*

Systems 2003: CoopIS, DOA, and ODBASE - OTM Confederated International Conferences, CoopIS, DOA, and ODBASE 2003, Catania, Sicily, Italy, November 3-7, 2003., volume 2888 of *Lecture Notes in Computer Science*, pages 407–425. Springer, 2003.

[RRD04] Stefanie Rinderle, Manfred Reichert, and Peter Dadam. Flexible support of team processes by adaptive workflow systems. *Distributed and Parallel Databases*, 16(1):91–116, 2004.

[RRMD09] Manfred Reichert, Stefanie Rinderle-Ma, and Peter Dadam. Flexibility in process-aware information systems. In Kurt Jensen and Wil M. P. van der Aalst, editors, *Transactions on Petri Nets and Other Models of Concurrency II, Special Issue on Concurrency in Process-Aware Information Systems*, volume 5460 of *Lecture Notes in Computer Science*, pages 115–135. Springer, 2009.

[SOS05] Shazia Wasim Sadiq, Maria E. Orlowska, and Wasim Sadiq. Specification and validation of process constraints for flexible workflows. *Information Systems*, 30(5):349–378, 2005.

[SOSF04] Shazia Sadiq, Maria Orlowska, Wasim Sadiq, and Cameron Foulger. Data flow and validation in workflow modelling. In *ADC '04: Proceedings of the 15th Australasian Database Conference*, pages 207–214, Darlinghurst, Australia, Australia, 2004. Australian Computer Society, Inc.

[Sti10] Stiftung für Hochschulzulassung. Das Studienplatzangebot zum Wintersemester 2010/11, 2010. Available from: http://www.hochschulstart.de/index.php?id=1802 [cited November 8, 2010].

[SZNS06] Sherry X. Sun, Leon J. Zhao, Jay F. Nunamaker, and Olivia R. Sheng. Formulating the data-flow perspective for business process management. *Information Systems Research*, 17(4):374–391, December 2006.

[TAS08] Nikola Trčka, Wil M. P. van der Aalst, and Natalia Sidorova. Analyzing control-flow and data-flow in workflow processes in a unified way. Technical Report No. 08–31, Technische Universiteit Eindhoven, 2008.

[TAS09] Nikola Trčka, Wil M. P. van der Aalst, and Natalia Sidorova. Data-flow anti-patterns: Discovering dataflow errors in workflows. In Pascal van Eck, Jaap Gordijn, and Roel Wieringa, editors, *Advanced Information Systems Engineering, 21st International Conference, CAiSE 2009, Amsterdam, The Netherlands, June 8-12, 2009. Proceedings*, volume 5565 of *Lecture Notes in Computer Science*, pages 425–439. Springer, 2009.

[Wes07] Mathias Weske. *Business Process Management: Concepts, Languages, Architectures*. Springer, 1st edition, 2007.

[Wor99] Workflow Management Coalition. *Workflow Management Coalition – Terminology & Glossary*. Workflow Management Coalition, February 1999. Document Number WFMC-TC-1011 – Document Status – Issue 3.0.

[WRMR11] Barbara Weber, Manfred Reichert, Jan Mendling, and Hajo A. Reijers. Refactoring large process model repositories. *Computers in Industry*, 62(5):467–486, 2011.

[WRRM08] Barbara Weber, Manfred Reichert, and Stefanie Rinderle-Ma. Change patterns and change support features – enhancing flexibility in process-aware information systems. *Data & Knowledge Engineering*, 66(3):438–466, 2008.

[WSR09] Barbara Weber, Shazia Wasim Sadiq, and Manfred Reichert. Beyond rigidity – dynamic process lifecycle support. *Computer Science – Research & Development*, 23(2):47–65, 2009.

[WW10] Matthias Weidlich and Mathias Weske. Structural and behavioural commonalities of process variants. In *2nd Central-European Workshop on Services and their Composition, Services und ihre Komposition, ZEUS 2010, Berlin, Germany, February 25–26, 2010. Proceedings*, volume 563 of *CEUR Workshop Proceedings*, pages 41–48, 2010.

Liste der bisher erschienenen Ulmer Informatik-Berichte

Einige davon sind per FTP von `ftp.informatik.uni-ulm.de` erhältlich
Die mit * markierten Berichte sind vergriffen

List of technical reports published by the University of Ulm

Some of them are available by FTP from `ftp.informatik.uni-ulm.de`
Reports marked with * are out of print

91-01 *Ker-I Ko, P. Orponen, U. Schöning, O. Watanabe*
Instance Complexity

91-02* *K. Gladitz, H. Fassbender, H. Vogler*
Compiler-Based Implementation of Syntax-Directed Functional Programming

91-03* *Alfons Geser*
Relative Termination

91-04* *J. Köbler, U. Schöning, J. Toran*
Graph Isomorphism is low for PP

91-05 *Johannes Köbler, Thomas Thierauf*
Complexity Restricted Advice Functions

91-06* *Uwe Schöning*
Recent Highlights in Structural Complexity Theory

91-07* *F. Green, J. Köbler, J. Toran*
The Power of Middle Bit

91-08* *V.Arvind, Y. Han, L. Hamachandra, J. Köbler, A. Lozano, M. Mundhenk, A. Ogiwara, U. Schöning, R. Silvestri, T. Thierauf*
Reductions for Sets of Low Information Content

92-01* *Vikraman Arvind, Johannes Köbler, Martin Mundhenk*
On Bounded Truth-Table and Conjunctive Reductions to Sparse and Tally Sets

92-02* *Thomas Noll, Heiko Vogler*
Top-down Parsing with Simulataneous Evaluation of Noncircular Attribute Grammars

92-03 *Fakultät für Informatik*
17. Workshop über Komplexitätstheorie, effiziente Algorithmen und Datenstrukturen

92-04* *V. Arvind, J. Köbler, M. Mundhenk*
Lowness and the Complexity of Sparse and Tally Descriptions

92-05* *Johannes Köbler*
Locating P/poly Optimally in the Extended Low Hierarchy

92-06* *Armin Kühnemann, Heiko Vogler*
Synthesized and inherited functions -a new computational model for syntax-directed semantics

92-07* *Heinz Fassbender, Heiko Vogler*
A Universal Unification Algorithm Based on Unification-Driven Leftmost Outermost Narrowing

92-08* *Uwe Schöning*
On Random Reductions from Sparse Sets to Tally Sets

92-09* *Hermann von Hasseln, Laura Martignon*
Consistency in Stochastic Network

92-10 *Michael Schmitt*
A Slightly Improved Upper Bound on the Size of Weights Sufficient to Represent Any Linearly Separable Boolean Function

92-11 *Johannes Köbler, Seinosuke Toda*
On the Power of Generalized MOD-Classes

92-12 *V. Arvind, J. Köbler, M. Mundhenk*
Reliable Reductions, High Sets and Low Sets

92-13 *Alfons Geser*
On a monotonic semantic path ordering

92-14* *Joost Engelfriet, Heiko Vogler*
The Translation Power of Top-Down Tree-To-Graph Transducers

93-01 *Alfred Lupper, Konrad Froitzheim*
AppleTalk Link Access Protocol basierend auf dem Abstract Personal Communications Manager

93-02 *M.H. Scholl, C. Laasch, C. Rich, H.-J. Schek, M. Tresch*
The COCOON Object Model

93-03 *Thomas Thierauf, Seinosuke Toda, Osamu Watanabe*
On Sets Bounded Truth-Table Reducible to P-selective Sets

93-04 *Jin-Yi Cai, Frederic Green, Thomas Thierauf*
On the Correlation of Symmetric Functions

93-05 *K.Kuhn, M.Reichert, M. Nathe, T. Beuter, C. Heinlein, P. Dadam*
A Conceptual Approach to an Open Hospital Information System

93-06 *Klaus Gaßner*
Rechnerunterstützung für die konzeptuelle Modellierung

93-07 *Ullrich Keßler, Peter Dadam*
Towards Customizable, Flexible Storage Structures for Complex Objects

94-01 *Michael Schmitt*
On the Complexity of Consistency Problems for Neurons with Binary Weights

94-02 *Armin Kühnemann, Heiko Vogler*
A Pumping Lemma for Output Languages of Attributed Tree Transducers

94-03 *Harry Buhrman, Jim Kadin, Thomas Thierauf*
On Functions Computable with Nonadaptive Queries to NP

94-04 *Heinz Faßbender, Heiko Vogler, Andrea Wedel*
Implementation of a Deterministic Partial E-Unification Algorithm for Macro Tree Transducers

94-05 *V. Arvind, J. Köbler, R. Schuler*
On Helping and Interactive Proof Systems

94-06 *Christian Kalus, Peter Dadam*
Incorporating record subtyping into a relational data model

94-07 *Markus Tresch, Marc H. Scholl*
A Classification of Multi-Database Languages

94-08 *Friedrich von Henke, Harald Rueß*
Arbeitstreffen Typtheorie: Zusammenfassung der Beiträge

94-09 *F.W. von Henke, A. Dold, H. Rueß, D. Schwier, M. Strecker*
Construction and Deduction Methods for the Formal Development of Software

94-10 *Axel Dold*
Formalisierung schematischer Algorithmen

94-11 *Johannes Köbler, Osamu Watanabe*
New Collapse Consequences of NP Having Small Circuits

94-12 *Rainer Schuler*
On Average Polynomial Time

94-13 *Rainer Schuler, Osamu Watanabe*
Towards Average-Case Complexity Analysis of NP Optimization Problems

94-14 *Wolfram Schulte, Ton Vullinghs*
Linking Reactive Software to the X-Window System

94-15 *Alfred Lupper*
Namensverwaltung und Adressierung in Distributed Shared Memory-Systemen

94-16 *Robert Regn*
Verteilte Unix-Betriebssysteme

94-17 *Helmuth Partsch*
Again on Recognition and Parsing of Context-Free Grammars:
Two Exercises in Transformational Programming

94-18 *Helmuth Partsch*
Transformational Development of Data-Parallel Algorithms: an Example

95-01 *Oleg Verbitsky*
On the Largest Common Subgraph Problem

95-02 *Uwe Schöning*
Complexity of Presburger Arithmetic with Fixed Quantifier Dimension

95-03 *Harry Buhrman,Thomas Thierauf*
The Complexity of Generating and Checking Proofs of Membership

95-04 *Rainer Schuler, Tomoyuki Yamakami*
Structural Average Case Complexity

95-05 *Klaus Achatz, Wolfram Schulte*
Architecture Indepentent Massive Parallelization of Divide-And-Conquer Algorithms

95-06 *Christoph Karg, Rainer Schuler*
Structure in Average Case Complexity

95-07 *P. Dadam, K. Kuhn, M. Reichert, T. Beuter, M. Nathe*
ADEPT: Ein integrierender Ansatz zur Entwicklung flexibler, zuverlässiger kooperierender Assistenzsysteme in klinischen Anwendungsumgebungen

95-08 *Jürgen Kehrer, Peter Schulthess*
Aufbereitung von gescannten Röntgenbildern zur filmlosen Diagnostik

95-09 *Hans-Jörg Burtschick, Wolfgang Lindner*
On Sets Turing Reducible to P-Selective Sets

95-10 *Boris Hartmann*
Berücksichtigung lokaler Randbedingung bei globaler Zieloptimierung mit neuronalen Netzen am Beispiel Truck Backer-Upper

95-12 *Klaus Achatz, Wolfram Schulte*
Massive Parallelization of Divide-and-Conquer Algorithms over Powerlists

95-13 *Andrea Mößle, Heiko Vogler*
Efficient Call-by-value Evaluation Strategy of Primitive Recursive Program Schemes

95-14 *Axel Dold, Friedrich W. von Henke, Holger Pfeifer, Harald Rueß*
A Generic Specification for Verifying Peephole Optimizations

96-01 *Ercüment Canver, Jan-Tecker Gayen, Adam Moik*
Formale Entwicklung der Steuerungssoftware für eine elektrisch ortsbediente Weiche mit VSE

96-02 *Bernhard Nebel*
Solving Hard Qualitative Temporal Reasoning Problems: Evaluating the Efficiency of Using the ORD-Horn Class

96-03 *Ton Vullinghs, Wolfram Schulte, Thilo Schwinn*
An Introduction to TkGofer

96-04 *Thomas Beuter, Peter Dadam*
Anwendungsspezifische Anforderungen an Workflow-Mangement-Systeme am Beispiel der Domäne Concurrent-Engineering

96-05 *Gerhard Schellhorn, Wolfgang Ahrendt*
Verification of a Prolog Compiler - First Steps with KIV

96-06 *Manindra Agrawal, Thomas Thierauf*
Satisfiability Problems

96-07 *Vikraman Arvind, Jacobo Torán*
A nonadaptive NC Checker for Permutation Group Intersection

96-08 *David Cyrluk, Oliver Möller, Harald Rueß*
An Efficient Decision Procedure for a Theory of Fix-Sized Bitvectors with Composition and Extraction

96-09 *Bernd Biechele, Dietmar Ernst, Frank Houdek, Joachim Schmid, Wolfram Schulte*
Erfahrungen bei der Modellierung eingebetteter Systeme mit verschiedenen SA/RT–Ansätzen

96-10 *Falk Bartels, Axel Dold, Friedrich W. von Henke, Holger Pfeifer, Harald Rueß*
Formalizing Fixed-Point Theory in PVS

96-11 *Axel Dold, Friedrich W. von Henke, Holger Pfeifer, Harald Rueß*
Mechanized Semantics of Simple Imperative Programming Constructs

96-12 *Axel Dold, Friedrich W. von Henke, Holger Pfeifer, Harald Rueß*
Generic Compilation Schemes for Simple Programming Constructs

96-13 *Klaus Achatz, Helmuth Partsch*
From Descriptive Specifications to Operational ones: A Powerful Transformation Rule, its Applications and Variants

97-01 *Jochen Messner*
Pattern Matching in Trace Monoids

97-02 *Wolfgang Lindner, Rainer Schuler*
A Small Span Theorem within P

97-03 *Thomas Bauer, Peter Dadam*
A Distributed Execution Environment for Large-Scale Workflow Management Systems with Subnets and Server Migration

97-04 *Christian Heinlein, Peter Dadam*
Interaction Expressions - A Powerful Formalism for Describing Inter-Workflow Dependencies

97-05 *Vikraman Arvind, Johannes Köbler*
On Pseudorandomness and Resource-Bounded Measure

97-06 *Gerhard Partsch*
Punkt-zu-Punkt- und Mehrpunkt-basierende LAN-Integrationsstrategien für den digitalen Mobilfunkstandard DECT

97-07 *Manfred Reichert, Peter Dadam*
$ADEPT_{flex}$ - Supporting Dynamic Changes of Workflows Without Loosing Control

97-08 *Hans Braxmeier, Dietmar Ernst, Andrea Mößle, Heiko Vogler*
The Project NoName - A functional programming language with its development environment

97-09 *Christian Heinlein*
Grundlagen von Interaktionsausdrücken

97-10 *Christian Heinlein*
Graphische Repräsentation von Interaktionsausdrücken

97-11 *Christian Heinlein*
Sprachtheoretische Semantik von Interaktionsausdrücken

97-12 *Gerhard Schellhorn, Wolfgang Reif*
Proving Properties of Finite Enumerations: A Problem Set for Automated Theorem Provers

97-13 *Dietmar Ernst, Frank Houdek, Wolfram Schulte, Thilo Schwinn*
Experimenteller Vergleich statischer und dynamischer Softwareprüfung für eingebettete Systeme

97-14 *Wolfgang Reif, Gerhard Schellhorn*
Theorem Proving in Large Theories

97-15 *Thomas Wennekers*
Asymptotik rekurrenter neuronaler Netze mit zufälligen Kopplungen

97-16 *Peter Dadam, Klaus Kuhn, Manfred Reichert*
Clinical Workflows - The Killer Application for Process-oriented Information Systems?

97-17 *Mohammad Ali Livani, Jörg Kaiser*
EDF Consensus on CAN Bus Access in Dynamic Real-Time Applications

97-18 *Johannes Köbler,Rainer Schuler*
Using Efficient Average-Case Algorithms to Collapse Worst-Case Complexity Classes

98-01 *Daniela Damm, Lutz Claes, Friedrich W. von Henke, Alexander Seitz, Adelinde Uhrmacher, Steffen Wolf*
Ein fallbasiertes System für die Interpretation von Literatur zur Knochenheilung

98-02 *Thomas Bauer, Peter Dadam*
Architekturen für skalierbare Workflow-Management-Systeme - Klassifikation und Analyse

98-03 *Marko Luther, Martin Strecker*
A guided tour through *Typelab*

98-04 *Heiko Neumann, Luiz Pessoa*
Visual Filling-in and Surface Property Reconstruction

98-05 *Ercüment Canver*
Formal Verification of a Coordinated Atomic Action Based Design

98-06 *Andreas Küchler*
On the Correspondence between Neural Folding Architectures and Tree Automata

98-07 *Heiko Neumann, Thorsten Hansen, Luiz Pessoa*
Interaction of ON and OFF Pathways for Visual Contrast Measurement

98-08 *Thomas Wennekers*
Synfire Graphs: From Spike Patterns to Automata of Spiking Neurons

98-09 *Thomas Bauer, Peter Dadam*
Variable Migration von Workflows in *ADEPT*

98-10 *Heiko Neumann, Wolfgang Sepp*
Recurrent V1 – V2 Interaction in Early Visual Boundary Processing

98-11 *Frank Houdek, Dietmar Ernst, Thilo Schwinn*
Prüfen von C–Code und Statmate/Matlab–Spezifikationen: Ein Experiment

98-12 *Gerhard Schellhorn*
Proving Properties of Directed Graphs: A Problem Set for Automated Theorem Provers

98-13 *Gerhard Schellhorn, Wolfgang Reif*
Theorems from Compiler Verification: A Problem Set for Automated Theorem Provers

98-14 *Mohammad Ali Livani*
SHARE: A Transparent Mechanism for Reliable Broadcast Delivery in CAN

98-15 *Mohammad Ali Livani, Jörg Kaiser*
Predictable Atomic Multicast in the Controller Area Network (CAN)

99-01 *Susanne Boll, Wolfgang Klas, Utz Westermann*
A Comparison of Multimedia Document Models Concerning Advanced Requirements

99-02 *Thomas Bauer, Peter Dadam*
Verteilungsmodelle für Workflow-Management-Systeme - Klassifikation und Simulation

99-03 *Uwe Schöning*
On the Complexity of Constraint Satisfaction

99-04 *Ercument Canver*
Model-Checking zur Analyse von Message Sequence Charts über Statecharts

99-05 *Johannes Köbler, Wolfgang Lindner, Rainer Schuler*
Derandomizing RP if Boolean Circuits are not Learnable

99-06 *Utz Westermann, Wolfgang Klas*
Architecture of a DataBlade Module for the Integrated Management of Multimedia Assets

99-07 *Peter Dadam, Manfred Reichert*
Enterprise-wide and Cross-enterprise Workflow Management: Concepts, Systems, Applications. Paderborn, Germany, October 6, 1999, GI–Workshop Proceedings, Informatik ’99

99-08 *Vikraman Arvind, Johannes Köbler*
Graph Isomorphism is Low for ZPP^{NP} and other Lowness results

99-09 *Thomas Bauer, Peter Dadam*
Efficient Distributed Workflow Management Based on Variable Server Assignments

2000-02 *Thomas Bauer, Peter Dadam*
Variable Serverzuordnungen und komplexe Bearbeiterzuordnungen im Workflow-Management-System ADEPT

2000-03 *Gregory Baratoff, Christian Toepfer, Heiko Neumann*
Combined space-variant maps for optical flow based navigation

2000-04 *Wolfgang Gehring*
Ein Rahmenwerk zur Einführung von Leistungspunktsystemen

2000-05 *Susanne Boll, Christian Heinlein, Wolfgang Klas, Jochen Wandel*
Intelligent Prefetching and Buffering for Interactive Streaming of MPEG Videos

2000-06 *Wolfgang Reif, Gerhard Schellhorn, Andreas Thums*
Fehlersuche in Formalen Spezifikationen

2000-07 *Gerhard Schellhorn, Wolfgang Reif (eds.)*
FM-Tools 2000: The 4th Workshop on Tools for System Design and Verification

2000-08 *Thomas Bauer, Manfred Reichert, Peter Dadam*
Effiziente Durchführung von Prozessmigrationen in verteilten Workflow-Management-Systemen

2000-09 *Thomas Bauer, Peter Dadam*
Vermeidung von Überlastsituationen durch Replikation von Workflow-Servern in ADEPT

2000-10 *Thomas Bauer, Manfred Reichert, Peter Dadam*
Adaptives und verteiltes Workflow-Management

2000-11 *Christian Heinlein*
Workflow and Process Synchronization with Interaction Expressions and Graphs

2001-01 *Hubert Hug, Rainer Schuler*
DNA-based parallel computation of simple arithmetic

2001-02 *Friedhelm Schwenker, Hans A. Kestler, Günther Palm*
3-D Visual Object Classification with Hierarchical Radial Basis Function Networks

2001-03 *Hans A. Kestler, Friedhelm Schwenker, Günther Palm*
RBF network classification of ECGs as a potential marker for sudden cardiac death

2001-04 *Christian Dietrich, Friedhelm Schwenker, Klaus Riede, Günther Palm*
Classification of Bioacoustic Time Series Utilizing Pulse Detection, Time and Frequency Features and Data Fusion

2002-01 *Stefanie Rinderle, Manfred Reichert, Peter Dadam*
Effiziente Verträglichkeitsprüfung und automatische Migration von Workflow-Instanzen bei der Evolution von Workflow-Schemata

2002-02 *Walter Guttmann*
Deriving an Applicative Heapsort Algorithm

2002-03 *Axel Dold, Friedrich W. von Henke, Vincent Vialard, Wolfgang Goerigk*
A Mechanically Verified Compiling Specification for a Realistic Compiler

2003-01 *Manfred Reichert, Stefanie Rinderle, Peter Dadam*
A Formal Framework for Workflow Type and Instance Changes Under Correctness Checks

2003-02 *Stefanie Rinderle, Manfred Reichert, Peter Dadam*
Supporting Workflow Schema Evolution By Efficient Compliance Checks

2003-03 *Christian Heinlein*
Safely Extending Procedure Types to Allow Nested Procedures as Values

2003-04 *Stefanie Rinderle, Manfred Reichert, Peter Dadam*
On Dealing With Semantically Conflicting Business Process Changes.

2003-05 *Christian Heinlein*
Dynamic Class Methods in Java

2003-06 *Christian Heinlein*
Vertical, Horizontal, and Behavioural Extensibility of Software Systems

2003-07 *Christian Heinlein*
Safely Extending Procedure Types to Allow Nested Procedures as Values
(Corrected Version)

2003-08 *Changling Liu, Jörg Kaiser*
Survey of Mobile Ad Hoc Network Routing Protocols)

2004-01 *Thom Frühwirth, Marc Meister (eds.)*
First Workshop on Constraint Handling Rules

2004-02 *Christian Heinlein*
Concept and Implementation of C+++, an Extension of C++ to Support User-Defined Operator Symbols and Control Structures

2004-03 *Susanne Biundo, Thom Frühwirth, Günther Palm(eds.)*
Poster Proceedings of the 27th Annual German Conference on Artificial Intelligence

2005-01 *Armin Wolf, Thom Frühwirth, Marc Meister (eds.)*
19th Workshop on (Constraint) Logic Programming

2005-02 *Wolfgang Lindner (Hg.), Universität Ulm , Christopher Wolf (Hg.) KU Leuven*
2. Krypto-Tag – Workshop über Kryptographie, Universität Ulm

2005-03 *Walter Guttmann, Markus Maucher*
Constrained Ordering

2006-01 *Stefan Sarstedt*
Model-Driven Development with ACTIVECHARTS, Tutorial

2006-02 *Alexander Raschke, Ramin Tavakoli Kolagari*
Ein experimenteller Vergleich zwischen einer plan-getriebenen und einer leichtgewichtigen Entwicklungsmethode zur Spezifikation von eingebetteten Systemen

2006-03 *Jens Kohlmeyer, Alexander Raschke, Ramin Tavakoli Kolagari*
Eine qualitative Untersuchung zur Produktlinien-Integration über Organisationsgrenzen hinweg

2006-04 *Thorsten Liebig*
Reasoning with OWL - System Support and Insights –

2008-01 *H.A. Kestler, J. Messner, A. Müller, R. Schuler*
On the complexity of intersecting multiple circles for graphical display

2008-02 *Manfred Reichert, Peter Dadam, Martin Jurisch,l Ulrich Kreher, Kevin Göser, Markus Lauer*
Architectural Design of Flexible Process Management Technology

2008-03 *Frank Raiser*
Semi-Automatic Generation of CHR Solvers from Global Constraint Automata

2008-04 *Ramin Tavakoli Kolagari, Alexander Raschke, Matthias Schneiderhan, Ian Alexander*
Entscheidungsdokumentation bei der Entwicklung innovativer Systeme für produktlinien-basierte Entwicklungsprozesse

2008-05 *Markus Kalb, Claudia Dittrich, Peter Dadam*
Support of Relationships Among Moving Objects on Networks

2008-06 *Matthias Frank, Frank Kargl, Burkhard Stiller (Hg.)*
WMAN 2008 – KuVS Fachgespräch über Mobile Ad-hoc Netzwerke

2008-07 *M. Maucher, U. Schöning, H.A. Kestler*
An empirical assessment of local and population based search methods with different degrees of pseudorandomness

2008-08 *Henning Wunderlich*
Covers have structure

2008-09 *Karl-Heinz Niggl, Henning Wunderlich*
Implicit characterization of FPTIME and NC revisited

2008-10 *Henning Wunderlich*
On span-P^{cc} and related classes in structural communication complexity

2008-11 *M. Maucher, U. Schöning, H.A. Kestler*
On the different notions of pseudorandomness

2008-12 *Henning Wunderlich*
On Toda's Theorem in structural communication complexity

2008-13 *Manfred Reichert, Peter Dadam*
Realizing Adaptive Process-aware Information Systems with ADEPT2

2009-01 *Peter Dadam, Manfred Reichert*
The ADEPT Project: A Decade of Research and Development for Robust and Fexible Process Support
Challenges and Achievements

2009-02 *Peter Dadam, Manfred Reichert, Stefanie Rinderle-Ma, Kevin Göser, Ulrich Kreher, Martin Jurisch*
Von ADEPT zur AristaFlow® BPM Suite – Eine Vision wird Realität "Correctness by Construction" und flexible, robuste Ausführung von Unternehmensprozessen

2009-03 *Alena Hallerbach, Thomas Bauer, Manfred Reichert*

Correct Configuration of Process Variants in Provop

2009-04 *Martin Bader*
On Reversal and Transposition Medians

2009-05 *Barbara Weber, Andreas Lanz, Manfred Reichert*
Time Patterns for Process-aware Information Systems: A Pattern-based Analysis

2009-06 *Stefanie Rinderle-Ma, Manfred Reichert*
Adjustment Strategies for Non-Compliant Process Instances

2009-07 *H.A. Kestler, B. Lausen, H. Binder H.-P. Klenk. F. Leisch, M. Schmid*
Statistical Computing 2009 – Abstracts der 41. Arbeitstagung

2009-08 *Ulrich Kreher, Manfred Reichert, Stefanie Rinderle-Ma, Peter Dadam*
Effiziente Repräsentation von Vorlagen- und Instanzdaten in Prozess-Management-Systemen

2009-09 *Dammertz, Holger, Alexander Keller, Hendrik P.A. Lensch*
Progressive Point-Light-Based Global Illumination

2009-10 *Dao Zhou, Christoph Müssel, Ludwig Lausser, Martin Hopfensitz, Michael Kühl, Hans A. Kestler*
Boolean networks for modeling and analysis of gene regulation

2009-11 *J. Hanika, H.P.A. Lensch, A. Keller*
Two-Level Ray Tracing with Recordering for Highly Complex Scenes

2009-12 *Stephan Buchwald, Thomas Bauer, Manfred Reichert*
Durchgängige Modellierung von Geschäftsprozessen durch Einführung eines Abbildungsmodells: Ansätze, Konzepte, Notationen

2010-01 *Hariolf Beth, Frank Raiser, Thom Frühwirth*
A Complete and Terminating Execution Model for Constraint Handling Rules

2010-02 *Ulrich Kreher, Manfred Reichert*
Speichereffiziente Repräsentation instanzspezifischer Änderungen in Prozess-Management-Systemen

2010-03 *Patrick Frey*
Case Study: Engine Control Application

2010-04 *Matthias Lohrmann und Manfred Reichert*
Basic Considerations on Business Process Quality

2010-05 *HA Kestler, H Binder, B Lausen, H-P Klenk, M Schmid, F Leisch (eds):*
Statistical Computing 2010 - Abstracts der 42. Arbeitstagung

2010-06 *Vera Künzle, Barbara Weber, Manfred Reichert*
Object-aware Business Processes: Properties, Requirements, Existing Approaches

2011-01 *Stephan Buchwald, Thomas Bauer, Manfred Reichert*
Flexibilisierung Service-orientierter Architekturen

2011-02 *Johannes Hanika, Holger Dammertz, Hendrik Lensch*
Edge-Optimized À-Trous Wavelets for Local Contrast Enhancement with Robust Denoising

2011-03 *Stefanie Kaiser, Manfred Reichert*
Datenflussvarianten in Prozessmodellen: Szenarien, Herausforderungen, Ansätze

Ulmer Informatik-Berichte
ISSN 0939-5091

Herausgeber:
Universität Ulm
Fakultät für Ingenieurwissenschaften und Informatik
89069 Ulm